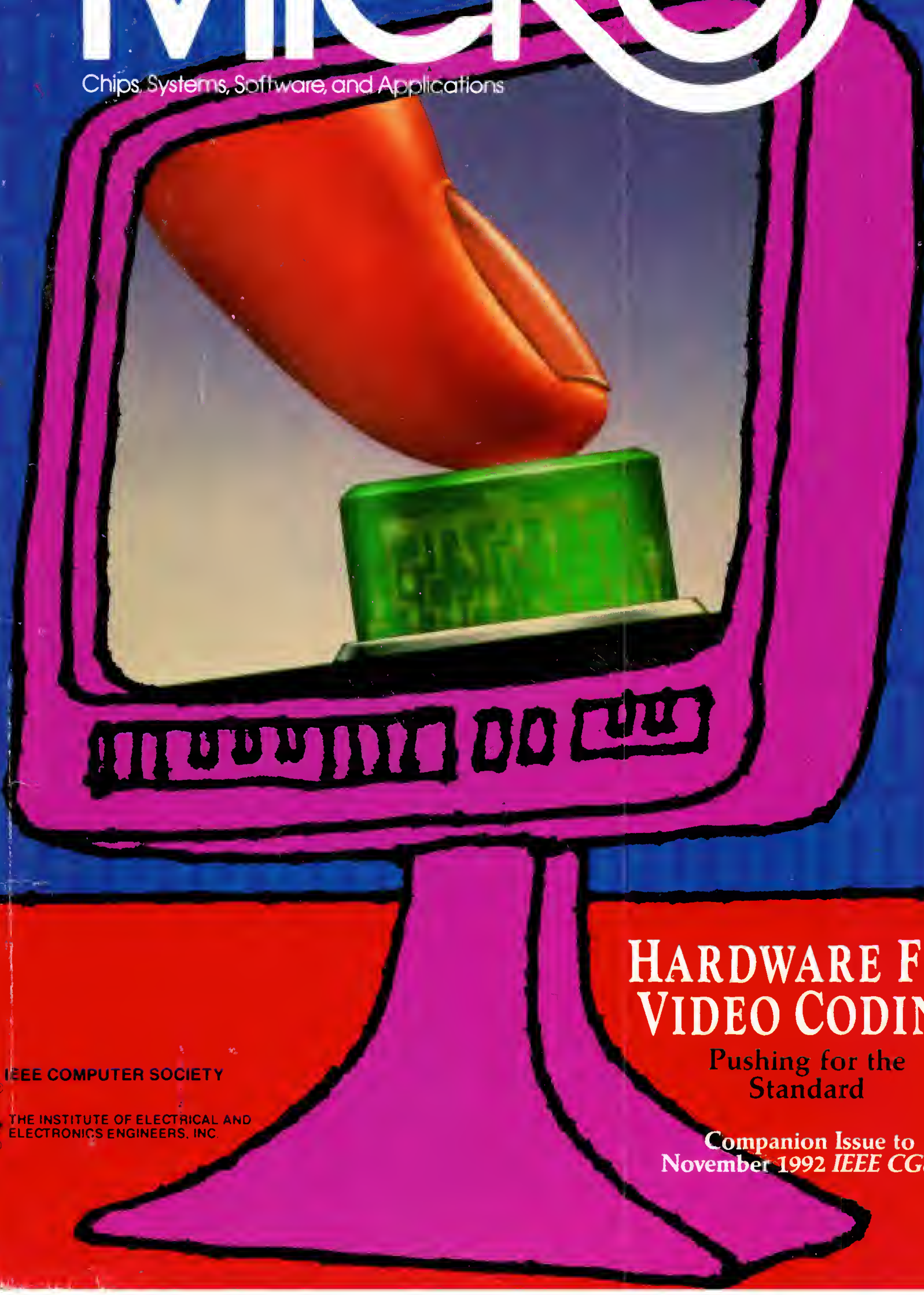
IEEE MICRO

Chips, Systems, Software, and Applications

OCTOBER 1992



HARDWARE FOR VIDEO CODING

Pushing for the Standard

Companion Issue to November 1992 *IEEE CG&A*

IEEE COMPUTER SOCIETY

THE INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS, INC.

IEEE MICRO

Published by the IEEE Computer Society

Volume 12 Number 5 October 1992

F E A T U R E S

Cover Image: Image Bank West
Cover design: Design and Direction

Circulation: *IEEE Micro* (ISSN 0272-1732) is published bimonthly by the IEEE Computer Society, PO Box 3014, Los Alamitos, CA 90720-1264; IEEE Computer Society Headquarters, 1730 Massachusetts Ave., NW, Washington, DC 20036-1903; IEEE Headquarters, 345 East 47th St., New York, NY 10017. Annual subscription: $23 in addition to IEEE Computer Society or any other IEEE society member dues; $39 for members of other technical organizations. This journal is also available in microfiche form.
Postmaster: Send address changes and undelivered copies to *IEEE Micro*, PO Box 3014, Los Alamitos, CA 90720-1264. Second-class postage is paid at New York, NY, and at additional mailing offices. Canadian GST#125634188.

DEPARTMENTS

IEEE Computer Society
PO Box 3014, Los Alamitos, CA 90720-1264
(714) 821-8380

* Submit six copies of all articles and special-issue proposals to Dante Del Corso, Dipartimento di Elettronica, Politecnico di Torino, C.so Duca degli Abruzzi, 24, 10129 Torino, Italy; phone +39-11-556-4044; Compmail: d.delcorso; Internet: delcorso@polito.it; Bitnet: delcorso@itopoli
or
** Ashis Khan, Mips Computer Systems, Inc., 950 DeGuigne Drive, Sunnyvale, CA 94086; Internet: ashis@mips.com.
Author guidelines for article submission are available from the West Coast office (address at left).

Micro Law

Penalties for reverse engineering

Richard H. Stern

Oblon, Spivak, McClelland, Maier & Neustadt, P.C.

1755 Jefferson Davis Highway, Suite 400

Arlington, VA 22202

The US Senate, on June 4, 1992, passed Senator Hatch's "Criminal Sanctions for Violation of Software Copyright" bill, S.893. The bill then moved to the House of Representatives, which recently held a hearing on the bill. Representatives of software publishers (notably, Word Perfect Corp.) favored passage of the Senate bill. A representative of the IEEE-USA's intellectual property committee testified against the bill in its present form.

According to the bill's proponents, such as Senator Hatch, S.893 will, if enacted into law (see Congressional Record, June 4, 1992):

- provide a strong tool for prosecutors to limit the growing problem of software piracy
- provide US software publishers with redress against large-scale commercially oriented copying of computer programs
- staunch $1.6-billion annual losses to publishers of PC software
- prevent "a decline in this vibrant and important sector of our economy," and
- protect consumers against defective copies of software and ineligibility for support.

The bill passed by the Senate provides that anybody who reproduces or distributes, over any six months, more than 10 infringing copies of one or more computer programs is guilty of a felony. Where the number of infringing copies is from 11 to 49, the penalty is up to two years in the federal penitentiary and/or a fine of up to $250,000. Where the number of copies is 50 or more, the penalty is up to five years of imprisonment and/or the same fine.[1] (See the box.)

It is unclear how the count is to be made, so that making one copy of all of Norton Utilities or some other extensive package might be sufficient to send the perpetrator away for a long time.

It is also unclear what kind of copy the bill deals with. Is loading a computer program into RAM the reproduction of a copy? Is putting a computer program into a file server attached by a LAN to 50 PCs reproducing or distributing 50 copies? The spokesperson for software publishers who testified in favor of the bill at the House hearing was not too specific about such details.

Even more problematical is what kind of activity the new felony law would sweep up. What is an infringing copy? According to some courts, taking the "look and feel" of another computer program is copyright infringement. According to the latest "command interface" opinion (*Lotus Development Corp.* v. *Borland International, Inc.*, 788 F. Supp. 78, D. Mass. 1992), Quattro infringes Lotus 1-2-3. It does so because if you punch the keystrokes </FR> in one of Quattro's modes, you get the same result that you get when you punch </FR> in 1-2-3. It follows, of course, that Phillipe Kahn would be a felon many times over, under S.893. (How many copies of Quattro do you think he willfully reproduced and distributed for financial gain in any six-month period?)

According to the court in *Sega* v. *Accolade*, disassembling a computer program and making a copy in diskette or hard disk, or on a printout, is a copyright infringement. Which brings us to you, reader.

Have you loaded a disassembled code into RAM 11 times over the last six months? Or printed it out? That can be one computer program 11 times, 11 computer programs one time each, or anything in between. If so, get ready to be fitted for a striped suit. (We need not even consider what happens if you have put some tainted code into a memory attached to a LAN.) Senator Hatch will be having an arrest warrant made out with your name on it. Expect the FBI to be paying a call on you.

Do you have kids who hack programs? Other people's programs? Do they put them on bulletin boards, resulting in 11 or more copies? If so, "What are you in for, kid?" could be answered with, "I hacked 11 programs last semester." For any resemblance to *Alice's Restaurant*, thank the software publishers' PAC.

Isn't it wonderful what a great job the incumbents are doing to get the government off our backs?

Damage control

Put aside your questions like, why do we need this bill at all? You can't do anything about bad legislation that way in Washington. This is a problem in process. The best that you can do is clean up S.893's act. If you clean it up enough, and its proponents are deprived of the benefit of their secret-agenda items (perhaps, to kill off reverse engineering), they may give up, and there may end up being no bill at all.

On the other hand, there might just possibly be no secret agenda. It may just be that whoever drafted this bill for Word Perfect and the other big software publishers simply did not know (or care) anything about computer programs. There is more just plain dumbness around than you may appreciate, especially when it comes to drafting laws. (The famous slogan of Hill staffers is "We don't have to understand these bills; we just have to write them." Or pass them.) If you clean it up, then, a bill will pass, but it will not be as bad as it would otherwise have been.

Therefore, the practical question is how can one best clean this bill up before it gets passed? Damage control is the most important thing.

First, you have to focus on the possible problems that the bill (see the box) would create, if it becomes law. What kind of unintended consequences could the bill have? What kind of *intended* consequences might it have that its proponents will not admit to? What specific language changes would draw the sting from the bill? That is,

prevent the really harmful consequences? How can that be done simply and concisely, so that you do not use up the patience and attention span of those involved?

Also, Congresspersons love consensus; being forced into controversy pains them. They would prefer, generally, to believe or pretend to believe that no secret agendas exist, and that it is just a matter of *wordsmithing* to bring everyone into harmonious agreement. This view suggests that a preferable approach in dealing with bad legislation is to take (or act as if you take) the proponents of the legislation at their word. Concede, for example, that they have a real problem out there with remorseless pirates really stealing $1.6 billion in sales from them, that software pirates threaten to terminate their vibrant and important creative contributions (to make Bill Gates walk the plank, or whatever), and that they do not intend to put hacker kids in the pen and intimidate reverse-engineering EEs.

Then, they can raise no legitimate objections to specific language that immunizes the hacker kids and reverse-engineering EEs. They are not going to say, "Hey, that was what we really wanted to suppress." At most, they will have to dream up some argument

S.893 amendment

As amended by Hatch bill S.893, the criminal copyright infringement section[1] of the US Criminal Code would provide:

"Any person who [infringes a copyright willfully and for purposes of commercial advantage or financial gain]—

(1) shall be fined not more than $250,000 or imprisoned for not more than five years, or both, if the offense—

... (C) involves the reproduction or distribution, during any 180-day period, of at least 50 copies infringing the copyright in one or more computer programs (including any tape, disk, or other medium embodying such programs); or

(D) is a second or subsequent offense ... where a prior offense involved a computer program ...;

(2) shall be fined not more than $250,000 or imprisoned for not more than two years, or both, if the offense—

... (C) involves the reproduction or distribution, during any 180-day period, of more than 10 but less than 50 copies infringing the copyright in one or more computer programs (including any tape, disk, or other medium embodying such programs); and

(3) shall be fined not more than $25,000 or imprisoned for not more than one year, or both, in any other case."

about the modification making it harder for them to get at real pirates. The bottom line on this point is that if you can come up with specific language proposals to eliminate the most harmful consequences of a bad bill, you will get much farther with the Congresspersons than you will by simply opposing the legislation in principle.

The problem with this approach, of course, is that it puts a premium on your ability as a soothsayer. You can tell bad legislation by the smell test, usually. But you do not always recognize all of the things wrong with it. If you put "except A, B, and C" into a bill, you will keep A, B, and C from happening. But if D lurks in there and you do not recognize it, the process of exclusion will be incomplete. You will fail to say "except D," and D can then happen. That is why writing in exceptions is not fail-safe, the way killing a bill is. But the way the legislative process works, exception-writing is far more feasible almost every time. Later, unfortunately, you will realize that "mistakes were made."

IEEE proposal

The IEEE spokesman's stated objection to the bill, in its present form, was that he objected to criminalizing (felonizing) IEEE members for engaging in ordinary engineering activities. These might be reverse engineering or being responsible for the distribution of code that some judge, under present amorphous copyright law standards, might consider to infringe a "nonliteral aspect" of a copyright (such as by unlawfully appropriating the look and feel of another computer program, or its command structure, as with Quattro and 1-2-3). The IEEE witness also objected to driving software engineering overseas and to criminalizing hacker kids. (Copies of this testimony are available from the witness.[2]) What is unknown is whether he foresaw all of the problems in S.893. Certainly, there is a major component of intimidation, and "chilling" competitive activity, because of the unpredictability of copyright law.

The IEEE-USA will propose a modified version of the bill to the House Judiciary Committee, which is considering the bill. You may want to provide your own input to your Congressperson, or the House Judiciary Committee. (Jack Brooks of Texas is the chair of the full committee; Bill Hughes of New Jersey chairs the intellectual property subcommittee.) If you find some more bodies buried in S.893, tell somebody; the indictment you prevent may be your own.

The IEEE-USA proposal will be along the following lines: First, felony status attaches only when there are 250 copies, not just 11; and, also, the value of the stolen software must be at least $50,000 (as measured by lost sales or ill-gotten gains, whichever is greater). Second, the 250 copies must be of tape, disk, or other physical, tangible, "nontransitory" character. (*Transitory* is a term that copyright law applies to TV signals and other short-term phenomena.)

Most important, the copies must "piratically infringe" the copyright. The newly defined term *piratically infringe* is based on S.893's stated purpose to suppress large-scale commercial software piracy. A piratical infringement of a computer program copyright, as defined in the IEEE-USA proposal, has the following characteristics: The copy is verbatim, or at least copies 50 percent of the instructions in the copyrighted computer program; if the computer program is reproduced or distributed in binary-coded format (machine-readable code), the percentage is determined by counting bits. The copying is intentional. The copied code amounts to at least 10 percent of the defendant's computer program. (Just copying into one's program a small amount of code that is dictated by function or is needed for interchangeability or compatibility is thus excluded from coverage at the outset by this rule.) Reverse engineering and disassembly are specifically excepted, as are computer programs made after reverse engineering or disassembly, where the final code is not a copy of the copyrighted code.

Did the IEEE-USA intellectual property group catch all of the potential problems? If you spot any more, speak up.

References

1. US Criminal Code, 18 U.S.C. § 2319(b).
2. David Ostfeld, 1-800-342-5829 (1-800-DialTax), chair, IEEE-USA intellectual property committe.

Reader Interest Survey

Indicate your interest in this department by circling the appropriate number on the Reader Service Card.

Low 177   Medium 178   High 179

IEEE MICRO

Reader Survey

HOW ARE WE DOING? We want to serve you better; tell us how by completing and returning the following by mail or fax to (714) 821-4010.

1. I read *IEEE Micro*:
- ❏ always
- ❏ often
- ❏ sometimes
- ❏ never

2. I find its scope of coverage:
- ❏ too narrow; should add areas of ______________________
- ❏ just right
- ❏ too broad; should cut areas of ______________________

3. Specific topics I would like to see covered in *IEEE Micro* next year are:

1. ______________________
2. ______________________
3. ______________________

4. I find the articles in *IEEE Micro*:
- ❏ are practically oriented
- ❏ are too theoretical
- ❏ have technical value
- ❏ point to new findings
- ❏ are too long
- ❏ need longer abstracts (now 50-75 words)
- ❏ need signposts on each page to highlight the content

5. I rank departments as follows: (circle appropriate number) Never read=1 Always read=5

Micro Law	1	2	3	4	5
Micro Review	1	2	3	4	5
Micro Standards	1	2	3	4	5
Micro News	1	2	3	4	5
New Products	1	2	3	4	5
On the Edge	1	2	3	4	5
Software Report	1	2	3	4	5

6. If I could change anything about *IEEE Micro*, I'd:

____

____

____

7. My issue is delivered:
- ❏ on time
- ❏ in good condition
- ❏ late
- ❏ damaged

8. I also read *IEEE Design & Test of Computers*:
- ❏ yes
- ❏ no

Other journals I read are: ______________________

____

____

9. My job title and application area are: ______________________

____

____

10. I hold the following degree(s): ______________________

____

11. I've submitted articles to *IEEE Micro*:
- ❏ yes
- ❏ no

continued on next page

12. I liked the following articles from the August 1991 to October 1992 issues:
(Check as many as necessary.)

August 1991 - Far East
- ❏ Toward advanced parallel processing
- ❏ Gmicro/100 32-bit microprocessor
- ❏ ITRON-MP adaptive real-time kernel
- ❏ Sure System 2000 communications processor
- ❏ KMDS expert system: integrated hardware/software design

October 1991 - General Interest
- ❏ Simulating a function of visual peripheral processes
- ❏ Computer analysis of the myoelectric signal
- ❏ Alphorn remote procedure call environment
- ❏ RISC processor for embedded applications in an ASIC
- ❏ Performance and the i860

December 1991 - Database Machines
- ❏ VLSI accelerators for large database systems
- ❏ Associative accelerator
- ❏ Fine-grain architecture for relational database operations
- ❏ Parallel, scalable database computer
- ❏ Rinda relational database processor

February 1992 - General Interest
- ❏ Scalable coherent interface and related standards
- ❏ Unix and the Am29000
- ❏ Hardware requirements for neural network classifiers
- ❏ Experimentation with hypercube database engines
- ❏ Conformance testing of VMEbus/Multibus II products

April 1992 - Hot Chips III
- ❏ The Mips R4000 processor
- ❏ Message-driven processor
- ❏ Motorola 88110 superscalar RISC
- ❏ Proposed SSBLT standard

June 1992 - Associative Processors and Memories
- ❏ Associative processors and memories survey
- ❏ Pattern-addressable memory
- ❏ Dynamic associative processor for machine vision
- ❏ Cascading content-addressable memories
- ❏ Associative processing module for vision

August 1992 - Microelectronics in Europe
- ❏ CMOS technology trends and economics
- ❏ Consumer applications
- ❏ A/D technologies for mixed-signal processing
- ❏ Trends in library development
- ❏ European activities for EDA standardization
- ❏ Floating-point processors for parallel processors
- ❏ Special report: MITI's Real-World Computing program

October 1992 - Video Coding Hardware
- ❏ Processing hardware for real-time coding
- ❏ Digital video coding techniques for HDTV
- ❏ A DSC-HDTV video decoder system
- ❏ A 160-Mpixel/s IDCT processor for HDTV
- ❏ Programmable Vision Processor/Controller
- ❏ V.42bis standard for data-compressing modems
- ❏ Supercomputer performance for neural net simulation with an array of DSPs

Return your comments by fax to (714) 821-4010 or mail to *IEEE Micro* Reader Survey, PO Box 3014, Los Alamitos, CA 90720-1264.

Software Report

ISDN progress; civil vs. military technology

David K. Kahaner

US Office of Naval Research, Far East

kahaner@cs.titech.ac.jp

Though the Integrated Services Digital Network is outside my expertise area, I am consistently asked about it by Western scientists. The following section on ISDN has been prepared mostly by the US Embassy staff in Tokyo, though it contains some added detail and editing of my own. The important points here are that broadband ISDN, high-speed communication, is coming to Japan. Companies are preparing products for the time when, not if, ISDN will be widely available. The actual date this reaches large numbers of Japanese subscribers is less important than the sense that it is moving inexorably forward. (Now digital telephone boxes are popping up in Tokyo, at least, with ISDN plugs for computer and fax.) NTT is the major agent, and its monthly *NTT Review* is full of articles about applications.

ISDN

The current narrowband Integrated Services Digital Network (N-ISDN) went into operation in Japan in April 1988 with the implementation of NTT's INS Net-64, which has a 64-Kbps transmission capability. In June 1989 NTT introduced INS Net-1500, with a much higher speed of 1.5 Mbps. INS Net-1500 allows multimedia communication, including teleconferencing, and it is possible to transmit a document page via fax in three seconds. INS Net started with 29 users and 114 subscriber lines. Total INS Net-64/1,500 subscriber lines have now passed the 60,000 mark. NTT claims that the number of contracts for ISDN circuits will reach 80,000, including about 2,000 areas in Japan.

With a maximum transmission speed of 1.5-Mbps, N-ISDN service is limited to the transmission of voice, low- and medium-speed data, still pictures, and simple moving images. Broadband ISDN (B-ISDN), with transmission speeds as high as 620 Mbps, will handle high-density media such as high-definition television (HDTV), cable TV, and videotex. The asynchronous transfer mode (ATM) technique, key to B-ISDN switching, increases both speed and frequency bandwidth by a new transmission protocol.

In N-ISDN, telephone, fax, video, and TV signals are divided and passed through several different switching systems and then recombined just before reaching the receiving terminals. ATM technology integrates these into a single net. The individual ATM transmitting terminal chops the information waves into cells of fixed lengths, assigns labels to them, and sends the "wavelets" to the net. When these cells arrive at the receiving end, the various information signals, grouped by assigned label, are directed to appropriate telephones, computers, or TV conference terminals.

Although ISDN is still in the fledgling stage, Japanese industry is busy preparing for the second-generation B-ISDN, which is up to 2,000 times faster than the existing ISDN. In 1990, the International Telegraph and Telephone Consultative Committee (CCITT) introduced formal guidelines for B-ISDN. The Japanese telecommunications provider, NTT, plans to have B-ISDN operational for commercial users in 1996 with three distinct features: ATM net, optical-fiber communication, and "opticalization" of components. The new technologies will begin to replace the existing ISDN infrastructure around the year 2000 and is planned to be completed by 2015. It is claimed that optical-fiber will reach cost parity with copper by 1995.

Toshiba is developing new 10-Gbps transmitting and receiving equipment for commercial availability in 1996. This equipment will use one optical fiber to carry 120 000 telephone lines simultaneously up to 80 km. It will feature several gallium arsenide ICs capable of processing information and claimed to be three to five times faster than conventional silicon ICs. NTT has also successfully carried out a 10-Gbps optical transmission experiment using dispersion-shifted, single-mode optical fibers with a combined length of 1,260 km, which are installed in a commercial route between Tokyo and Hamamatsu (route length is 326 km).

NTT's Large-Scale Integration (LSI) Laboratory recently introduced two new types of LSI chips for use with optical communications in B-ISDN. NTT intends the FIFO (first-in, first-out) LSI chip for ATM use, and the time-slot-converter LSI chip for use in circuit divisions and multiplexing. The FIFO LSI chip uses 0.8-micron BiCMOS technology on a single chip to upgrade processing speeds from the 80 Mbps of conventional FIFO to 250 Mbps, while reducing power consumption from 20W to 0.8W.

The NTT system also involves an optical switching system based on NTT's VSTEP (vertical-to-surface-transmission electrophotonic device) technology. The prototype switching system includes a superhigh-speed, 4×4 (four inputs, four outputs) cell-fluting circuit. This circuit makes it possible to switch optical signals without converting them to electrical signals, and an optical buffer memory that holds input signals until they are placed on the output lines.

NTT's experiments, said to be the first of their kind, verified the feasibility of high-speed optical throughput switching, required to achieve 1-Tbps ATM switching for B-ISDN. In the area of standardization, NTT has developed a B-ISDN-quality standardization system called Square (Subjective Quality Assessment Reference) capable of measuring and standardizing both sound and video quality through simulation. NTT is seeking to establish Square as an international standard for B-ISDN quality control through the CCITT.

Fujitsu has developed FLM2400, a synchronous digital hierarchy (SDH) optical telecommunications device for B-ISDN with a transmission speed of 2.4 Gbps. NTT also claims to be working in this area. SDH makes it possible to directly multiplex and cross-connect channels that have different capacities, allowing greater freedom and operational flexibility.

Slowing down HDTV signals could provide an effective means of "time sharing" among various signals while reducing data loss or contamination.

In the B-ISDN-compatible terminals arena, Fujitsu has developed Monster, a multimedia workstation with a image processing capability. It has also announced a plan to produce an HDTV signal compression system within two or three years that could vary transmission speeds over a B-ISDN line by 60 Mbps to 130 Mbps. Since HDTV signals require extremely high transmission rates (approaching 1 Gbps), the optical-fiber line over which the signals are sent can become too crowded for other data transmissions to take place concurrently. The ability to slow down the HDTV signals could provide an effective means of "time sharing" among various signals while reducing data loss or contamination, a major problem in high-speed ATM transmissions.

Civil vs. military technology

This section on Japan's consumer demand points up the importance of this market as a key driving force for the country's progress in high technology. In *Techno Japan* (Vol. 24, No. 9, Sept. 1991) this point is stated and amplified even further. While some of the points of logic made in this lightly edited report may not follow, its conclusions, at least as viewed by the Japanese, are inescapable: Consumer market forces are ultimately more successful than military in the development of state-of-the-art technology. (Don't read this for perfect inferencing, only for a clear statement of philosophy.)

Military technology. Many say that advanced technologies always lie with war machines, or that they have progressed by war machine development. When a nation or a group of people fights for survival, victory or defeat is vitally dependent on superiority of weapons. As such military strength of a nation has vital impact on a nation's fate, every nation expands vast efforts on development of military technology.

Efforts to develop advanced military technology have been made with utmost priority by many countries. This movement generates a stance that development of advanced technology should focus on military technology, and civil technology should be fostered by transferring technology from the military.

The stance that military technology should lead the advanced technology of a nation may have a reasonable basis from a short-term viewpoint. Just before World War II, Japanese military technology competed with US technology, for instance, in battleship building, designing of some kinds of fighters, lens polishing, and so on. Development of these technologies was promoted as

continued on p. 69

Guest Editors' Introduction

Processing Hardware for Real-Time Video Coding

Gilles Privat

France Telecom CNET-Grenoble

Eric Petajan

AT&T Bell Labs

Visual communication lies at the hub of the information technology triangle where telecommunication, computing, and media industries interpenetrate. This convergence, fostered by pervasive digital technology, will ideally bring about seamless integration of a highly diverse range of applications. Presently, this range is splintered along a throughput/resolution spectrum, with very different states of development.

This special issue focuses on full-motion digital video, excluding still image, analog video, and low-end software-based digital video solutions such as animated JPEG. (See the box on the next page for a glossary of applicable terms.) Applications fall into three broad categories: two-way interpersonal *transmission*, one-way *distribution* (broadcast and cable), and *storage/retrieval*. All these application domains have specific requirements concerning coder-vs.-decoder complexity, robustness, random access, processing delay, and so on. Whatever the transmission or storage medium may be, all these applications actually rely on source and channel coding-decoding systems, so that they may be conceptually unified under digital videocommunication.

The applications include the following:

- **ISDN videophones** got an early start with the 1990 finalization of the H.261 (p×64) standard.[1] These videophones now face competition from both non-ISDN low-end products and high-end integrated multimedia systems.
- **Videoconference** systems have been in operation for more than 10 years and still represent the primary application of digital video transmission, relying mainly, however, on proprietary solutions. The future of videoconferencing probably lies in an integration with standard multimedia environments, including CCITT p×64 (H.261) as a common denominator.
- The over-hyped emergence of **multimedia**[1,2] computing systems has spawned the development of numerous would-be de facto standards for digital video, put forward by most of the key players in this highly competitive field (Philips CDI, Intel DVI, Apple Quick Time, Commodore CDTV, Sony-Microsoft CD-ROM XA). Commercial systems might include proprietary enhancements and be multistandard, using the formal international MPEG-1 norm as a fall-back standard to ensure minimal interoperability.
- Video for high-end multimedia systems (above 1.5 Mbps) converges with **digital TV**, as partially addressed by MPEG-2. This standard in progress attempts to fill the gap with already-existing standards for studio- and contribution-use digital TV (CCIR 601 and CMTT/2, 34-45 Mbps). Digital TV distribution on broadband ISDN networks could open up new media applications (video on demand, news on demand, interactive TV).
- Secondary distribution of **HDTV** by terrestrial broadcast and satellite involves the issue of backward compatibility and the controversial analog-vs.-digital choice, with no agreement on a worldwide standard to date. E. Petajan discusses the status of US proposals in a companion article. Digital coding of HDTV for contribution and pri-

Companion issue to November 1992 *IEEE Computer Graphics & Applications* on ICs for computer graphics and imaging. See p. 40 for more information.

Glossary

ATM	Asynchronous transfer mode, the unified switching protocol adopted for broadband ISDN networks that will make it possible to support variable bit rate video transmission, along with other services of various characteristics
CCITT	International Telegraph and Telephone Consultative Committee
CDI	Compact Disc Interactive
CD-ROM XA	Extended architecture for CD-ROM storage of audio and video information in a digital format
CCIR	International Radio Consultative Committee
Channel coding	Coding of a signal to minimize error probability when passing through a given transmission/storage medium
CMTT	Mixed Television and Telecommunications Committee (a joint committee between CCIR and CCITT)
Contribution link	A link for professional use that makes it possible to further process the video signal in digital form
DCT	Discrete cosine transform for coding compression
DVI	Digital Video Interactive
Forward/backward compatibility	Compatibility of new receivers with signals encoded in former standards, or of new transmission standards with receivers of an earlier generation. For TV and HDTV, simulcast in the US and the MAC family in Europe address backward compatibility.
H.261	CCITT recommendation for ISDN specifies a common method for visual telephony. Also known an p×64 (Kbps, with p = 1 ... 30).
IEC	International Electrotechnical Commission
ISDN	Integrated Services Digital Network, an international telecommunications service carrying simultaneous voice/video/data
JPEG	ISO/CCITT Joint Photographic Experts Group sponsoring a general-purpose still-image compression standard
MPEG	ISO/IEC Moving Picture Experts Group sponsoring generic video/audio compression standards 1 and 2
Primary distribution	Transmission between the studio and the final HF transmitter
Secondary distribution	Transmission between HF transmitter and the end user
Source coding	Coding of an information source (here, the video signal) with the most compact possible digital representation. The rate of this minimal code corresponds to the information content or entropy of the source.
Upward/downward compatibility	Compatibility along a hierarchy of digital standards with different formats. If, for example, a digital TV signal is embedded in an HDTV signal with twice the same spatial resolution, TV receivers should be capable of decoding a relevant part of the HDTV signal (downward compatibility) and vice versa for upward compatibility.
VSP	Video signal processor

mary distribution has also been studied independently for quite some time in Europe and Japan. The main focus of long-term research worldwide is toward full digital HDTV distribution on ATM-based broadband ISDN networks, with strong emphasis on variable bit-rate coding and upward and downward compatibility among all digital video services. Future digital **SHD** (super high-definition) imaging systems target the requirements of medical, publishing, CAD, and entertainment application, with resolutions up to 4,096×4,096 bits. They might become the common multiple that could unite all present-day digital and analog formats, including 35-mm film.

Algorithmic issues

The development of digital video applications proceeded hand in hand with that of compression algorithms. The established or future p×64 (H.261), MPEG-1, and MPEG-2 standards all use some mixture of transform intraframe coding (see, for example, Rabbani and Jones[3]) and predictive or interpolative interframe coding, for spatial and temporal redundancy reduction. The Petajan overview article surveys some of these classical techniques as they are being applied to digital HDTV in the US. These first-generation algorithms still dominate the industrial image compression arena, targeting compression ratios in the 5-100 range. They have been incrementally improved for more than 20 years, and the last entropy bit is close to having been squeezed from them. Less mainstream is the region-based coding with vector quantization used by the Intel DVI. Vector quantization-based systems have also been proposed for satellite and cable digital TV transmission (Scientific Atlanta) and videoconferencing (Picture Tel).

An area of active research is in so-called hierarchical techniques for compatible coding of scalable images at different resolutions,[4] for which pyramid and subband techniques are being actively investigated. Subband and transform representations are actually quite close in principle, and some subband systems proposed for the MPEG-2 competition in November 1991[5] performed very closely to classical DCT-based schemes.

Emerging model-based techniques (see Forchheimer and Kronander[6]) exploit more directly the structural relevance of the image contents and make it possible to improve compression ratios by at least an order of magnitude. These techniques radically move away from classical waveform-based information-theoretic coding, coming closer instead to vision (for encoding/analysis) and computer graphics rendering (for decoding/synthesis). They trade compression ratio for visual quality and generality of the decoding system. Fractal-based methods are among these promising approaches that identify the parameters of the image according to a prespecified model. Higher level models (face models), assuming a priori knowledge of the image contents, are being studied for application to videophone and videoconference coding. The resolution-independence of these parametric encodings points toward future universal high-level representations, freed from the rasterizations and sampling frequency constraints originating from the analog world.

VLSI hardware for video coding

The core of a video-communication system is the source coder-decoder, and we will mostly be concerned with this in the present issue. Channel coding also involves, however, challenging hardware problems, especially for satellite and terrestrial broadcast.

As the computational power required by most video coding algorithms operating in real time is on the order of several billion operations per second, actual application of these techniques has been made possible only by the progress in computing hardware. Their future market acceptance will depend on the availability of low-cost dedicated engines. Hardware for video coding is currently a very active field of both research and development. A wide spectrum of architectures has been addressed, ranging from dedicated programmable processors and parallel machines to fixed-function special-purpose hardware. All these systems have in common the extensive use of pipelining, the only means to match the real-time throughput constraint.

Fixed-function VLSI chips capture the full potential of parallel architectures and target the high-end application range. Most current offerings are chip sets, with each chip implementing a building block of a whole codec system. These solutions retain some level of flexibility to adapt to different coding algorithms and to the potential evolution of these algorithms. The articles by Ruetz and Tong (LSI Logic) and O. Duardo et al. (AT&T) also in this issue give examples of this approach. Such building blocks have also been designed by SGS-Thomson, Motorola, Zoran, NEC, and Prism (to mention only some of those who have addressed a complete coder this way, rather than isolated building blocks). Nonprogrammable single-chip solutions are available for JPEG-style, still-image coding (from C-Cube, SGS-Thomson, and others). In the near future, if bottom-line standards finally achieve worldwide acceptance, fully dedicated and streamlined fixed-function chips integrating a complete codec might appear for some low-end, single-standard digital video systems.

Programmable processors emphasize maximum flexibility for the implementation of multiple standards and adaptation to their present and future evolution. Early general-purpose VSP circuits of this kind did not generally offer enough computational horsepower to implement the most computation-intensive coding algorithms. A new generation of these VSPs, tailored and optimized for the specific requirements of video coding applications, has appeared. They offer enhanced parallelism and pipelining capabilities, such as exemplified in the article by D. Bailey et al. (IIT) in this issue. This promising approach has been adopted by many vendors as well as by system manufacturers for proprietary use, among which are Philips (VSP), Intel (i750 DVI processor), NTT (IDSP), NEC (S-VSP), Mitsubishi (DISP), Matsushita (VDSP), and C-Cube Microsystems (CL 950). Less mainstream is the Datawave circuit from ITT Intermetall which implements a fine-grained MIMD architecture with a mesh array of 16 processing elements on a single chip.[8]

Most current coding schemes are highly asymmetric in that coding is much more computation-intensive than decoding. This happens to fit with the requirements of most distribution and storage applications in which only decoding is required by end users and is thus much more cost-critical than coding. Encoding still benefits from close-to-real-time performance, though not necessarily with a highly integrated hardware. Parallel computers, either general-purpose or video-oriented machines, have been widely used for prototype video coding system development by Philips for CD-I (POOMA[1]), Intel for DVI (Meiko and iPSC/2),[1] NTT (NOVI-II HiPIPE for SHD), SEL (Standard Elektrik Lorenz), MIT, CCETT, and many others.

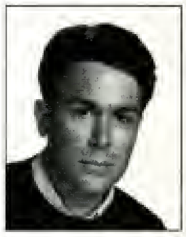
THE FUTURE ASSUREDLY LIES IN THE INTEGRATION of all video standards, possibly using some kind of resolution-independent, temporally and spatially scalable intermediate representation. The study of a so-called open architecture[7] for digital video that would decouple input, output, and transmission resolutions is currently very active. This trend could ultimately shift the emphasis in hardware implementations from today's rasterized sequential input toward massively parallel electronic retina-like circuits that move processing closest to where the image data is naturally available in parallel. These circuits have long been an active area of research for vision applications, but their potential use for coding might spur a renewed large-scale industrial interest.

Obviously, the field of hardware for video coding continues to evolve very quickly, and many new and possibly unforeseen developments are due to come about in the near future.

Be sure not to miss further articles that will appear in *IEEE Micro* on this exciting subject!

References

1. E.A. Fox, ed., Special Issue on Multimedia Systems, *Commun. ACM*, Vol. 34, No. 4, Apr. 1991, all pages.
2. Special Issue on Multimedia, *Computer*, Vol. 24 , No. 10, Oct. 1991, all pages.
3. M. Rabbani and P.W. Jones, *Digital Image Compression Techniques*, SPIE Optical Engineering Press, San Diego, Calif., 1991.
4. L. Vandendorpe, "Hierarchical Transform and Subband Coding of Video Signals," *Image Communication*, Vol. 4, No. 3, June 1992, pp. 245-262.
5. *Description of the VADIS-A3 Common Scheme, MPEG-2 Proposal 9*, MPEG-2 Group (ISO/CCITT), available from CCETT, Rennes, France, Nov. 1991.
6. R. Forchheimer and T. Kronander, "Image Coding—From Waveforms to Animation," *IEEE Trans. Acoustics, Speech and Image Processing*, Vol. 37, No. 12, Dec. 1989, pp. 2008-2023.
7. R.J. Kurgen, "Digital Video," *IEEE Spectrum*, Mar. 1992, pp. 24-30.
8. U. Schmidt and K. Caesar, "Datawave: A Single-Chip Multiprocessor for Video Applications," *IEEE Micro*, Vol. 11, No. 3, June 1991, pp. 22-25, 88-94.



Gilles Privat, a research engineer with CNET-Grenoble, a France Telecom research center, heads a research group working in the areas of parallel algorithms and VLSI architecture for image coding. His main activities include silicon compilation, VLSI design, VLSI signal processing, systolic architectures, computer arithmetic, algorithms and architectures for audio and video source coding, automata networks, and parallel processing.

Privat received engineering and doctoral degrees in signal and systems theory from ENST (Telecom Paris Institute). He is a member of the Institute of Electrical and Electronics Engineers, the Computer Society, and a member of the Editorial Board of *IEEE Micro*.

Eric Petajan is a supervisor in the Computer Technology Research Laboratory at AT&T Bell Labs, Murray Hill, New Jersey. His research activities include HDTV algorithms and architectures, interactive graphics, and visual speech processing (automatic lipreading). He also is an active participant in the US HDTV standardization process.

Petajan received his BS in physics from the University of Utah, MS degree in physics from the University of Illinois, and PhD in electrical engineering from the University of Illinois, Urbana-Champaign.

Direct questions regarding this special issue to Gilles Privat, CNET, BP 98, 38243 Meylan Cedex, France; or e-mail at privat@cns.cnet.fr.

Reader Interest Survey

Indicate your interest in this article by circling the appropriate number on the Reader Service Card.

Low 150	Medium 151	High 152

Digital Video Coding Techniques for US High-Definition TV

Motion-compensated transform coding is proposed for video compression in systems for the US terrestrial broadcast of HDTV. This noncritical overview of the video coding techniques used in the four proposed digital HDTV systems also provides basic descriptions of the discrete cosine transform, motion estimation and compensation, adaptive quantization, variable-length coding, and compressed video data formatting.

Eric Petajan

AT&T Bell Laboratories

Simulation and practical demonstrations of over-the-air transmission show that digital high-definition television is practical for terrestrial broadcasting in the United States. Presently, four digital HDTV systems are competing for the US terrestrial broadcast standard. A consortium between the Massachusetts Institute of Technology and General Instrument Corporation developed the Digicipher[1] and Channel Compatible Digicipher[2] (CCDC) systems. Zenith and AT&T offer the Digital Spectrum Compatible[3] (DSC) system, and a consortium of Philips Consumer Electronics Company, Thomson Consumer Electonics, David Sarnoff Research Center, National Broadcasting Company, and CLI support the Advanced Digital Television[4] (ADTV) system. The video coder in the ADTV system is based on the MPEG-1[5] proposed standard. I offer an overview of these systems but do not critically compare them for readers. (See the box for a glossary of terms.)

System requirements

The US Federal Communications Commission will allocate 6-MHz terrestrial broadcast channels for HDTV. Most of these channels are NTSC-taboo channels that contain interference from distant NTSC transmitters in the same band or adjacent channel interference from nearby transmitters. As a result, the FCC requires that the selected HDTV system provide acceptable viewer coverage without causing excessive interference with frequency collocated or adjacent NTSC channels.

The selected HDTV system should provide greatly improved picture quality compared to NTSC. The spatial resolution should be at least twice that of NTSC, horizontally and vertically, without exhibiting interlace artifacts. In addition, the HDTV system standard must avoid the chrominance artifacts and poor chrominance fidelity associated with NTSC. Finally, HDTV will have a 16×9 aspect ratio for compatibility with film (compared to 4×3 for NTSC). These bandwidth and picture quality requirements demand the use of video compression techniques with the highest possible performance.

Receiver complexity should be minimized to ensure that manufacturers can eventually provide consumers with receivers at a cost that will not inhibit the proliferation of HDTV. This constraint implies a preference for video coding and transmission techniques that place most of the complexity in the encoder rather than the decoder.

Color space transformation

Human psychovisual research[6] indicates that our ability to resolve spatial detail is much greater in the light intensity or luminance portion of the visual signal than in its isolated color or chrominance portion. Current color cameras and displays interface to red, green, and blue (RGB)

Glossary

ADTV	Advanced Digital Television HDTV system using 1,050-line interlaced scanning
CCDC	Channel Compatible Digicipher HDTV system using 787-line progressive scanning
DCT	Discrete cosine transform
DFD	Displaced frame difference or prediction error
Digicipher	HDTV system using 1,050-line interlaced scanning
DSC	Digital Spectrum Compatible HDTV system using 787-line progressive scanning
FCC	US Federal Communications Commission
HDTV	High-definition television
Intraframe compression	Uses only information in the current frame without temporal prediction
ISO	International Standards Organization
Inverse quantization	Mapping of coded quantization levels to original prediction levels
MPEG	Moving Picture Experts Group standard ISO/IEC JTC1/SC29/WG11
MPEG-1	Informal name of proposed MPEG standard DIS 1172
NTSC	US National Television System Committee
PAL	European analog video standard
Quantization	Mapping ranges of amplitude values to discrete levels
SECAM	European analog video standard
VLC	Variable-length code assigning short code words to frequently occurring values for lossless data compression

intensity signals. All HDTV systems transform the RGB signals to luminance and two chrominance signals to compress the chrominance more than the luminance. In the digital HDTV systems this transformation takes place in the spatial domain through decimation and in the spatial frequency domain by coarse quantization. The luminance and chrominance signals are transformed to RGB signals after decoding and before display.

Progressive vs. interlaced scanning

Psychovisual research[7] also indicates that loss of spatial resolution in moving areas of a picture is less visible than in stationary areas. Interlaced scanning is an attempt to take advantage of this for compression by spatiotemporal subsampling, as shown in Figure 1. This video compression technique was attractive for early TV systems (NTSC, PAL, and SECAM) because it did not require video data buffering. Also, higher static spatial resolution could be achieved for most pictures given the bandwidth limitations of early cameras and displays.

However, spatial and temporal aliasing causes annoying artifacts for certain types of vertical detail (computer-generated images). In this case, the effective vertical resolution must be greatly lowered by filtering to minimize the appearance of aliasing artifacts. Two of the proposed HDTV systems (DSC and CCDC) use progressive scanning with 720 active lines per frame (1/59.94th second). The other two systems (ADTV and Digicipher) use interlaced scanning with 480 lines per field (1/59.94th second) or 960 lines per frame (1/29.97th second). The primary motivation to use interlaced scanning today is the limited availability of progressive scan cameras in the short term.

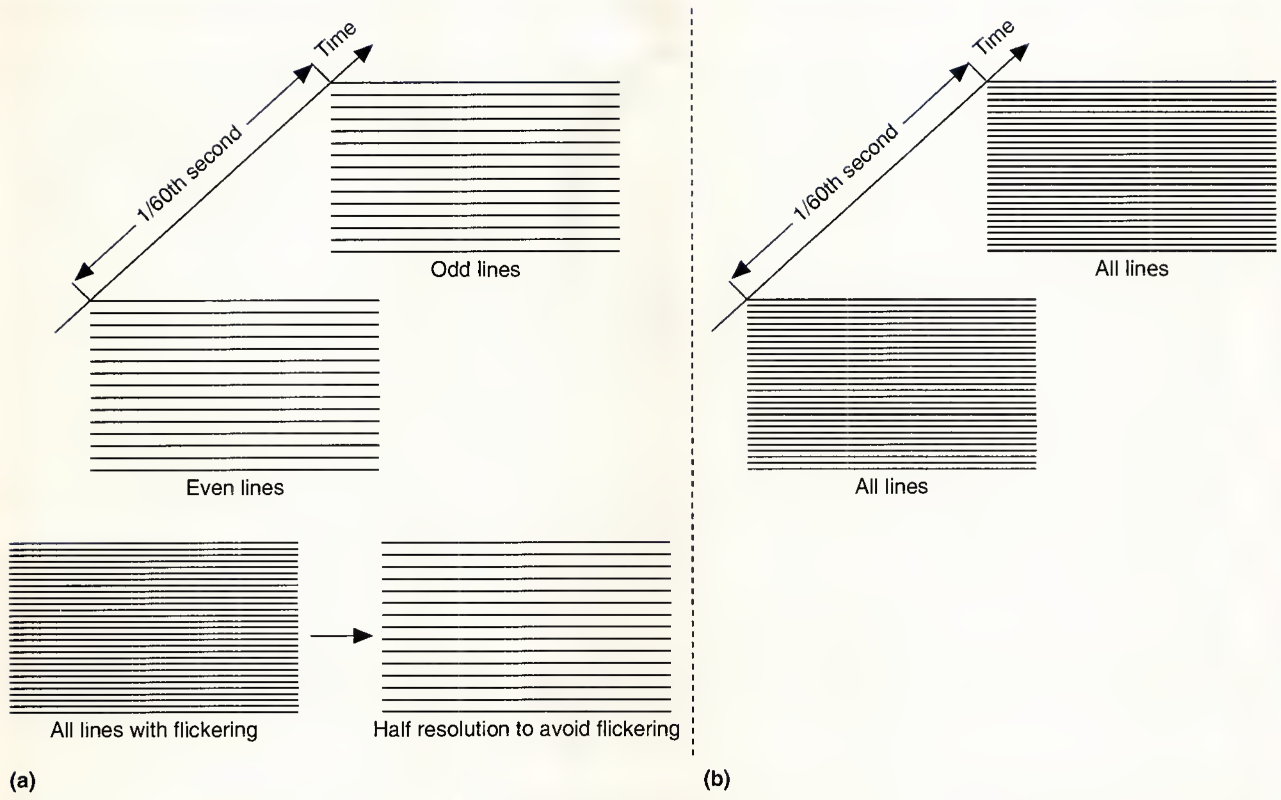


Figure 1. Interlaced vs. progressive scanning.

Frequency transformation

The primary compression technique used in all four digital HDTV systems transforms blocks of pixels into a quasifrequency domain followed by adaptive transform coefficient quantization. This technique transforms 8×8 blocks of pixels into 8×8 blocks of transform coefficients using the discrete cosine transform (DCT).[8]

The DCT representation of most pictures contains a majority of low-amplitude coefficients that can be either coarsely quantized or zeroed without significant visible distortion. In general, higher order (frequency) coefficients can be much more coarsely quantized than low-order coefficients. I describe the selection and quantization of DCT coefficients in a later section.

The successive application of two one-dimensional cosine transform operations changes a block of pixels into a block of coefficients. The separability of the 2D cosine transform reduces the number of multiplications required to implement the 2D transform. Denoting the 8×8 block of pixels by $[x]$ and presenting the 1D cosine transform matrix with [C], the 2D transformation of the block is given by

$$[X]^T = [C]\{[C][x]\}^T$$

The elements of matrix [C] are given by:

$$C_{kj} = \frac{2d(k)}{N}\cos\left[\frac{2j+1}{2N}\,xk\pi\right] \quad k, j = 0\ldots N-1$$

$$d(k) = \begin{cases} \dfrac{1}{\sqrt{2}} & k = 0 \\ 1 & k = 1\ldots N-1 \end{cases}$$

Motion compensation

Each digital HDTV system removes temporal redundancy by motion estimation and motion compensation.[2] As seen in Figure 2 on the next page, a system computes motion vectors between frames in the encoder and transmits them to the decoder. The DSC and ADTV systems compute motion vectors between original frames, while the GI/MIT systems

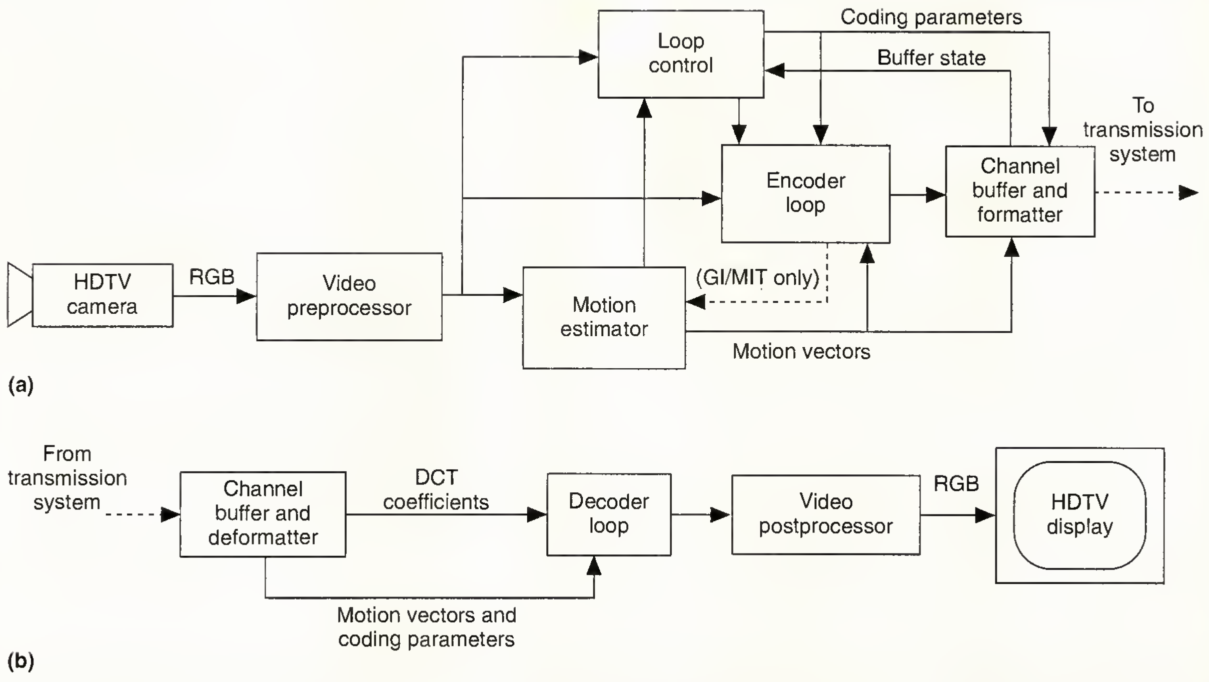


Figure 2. Video encoder (a) and decoder (b).

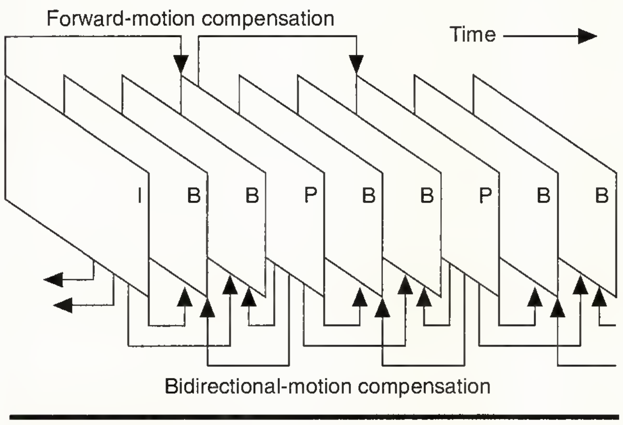

Forward-motion compensation
Time
I
B
B
P
B
B
P
B
B
Bidirectional-motion compensation

Figure 3. MPEG-1 group of pictures.

use the reconstructed previous frame and the current original frame to compute motion vectors.

The ADTV system does not perform motion compensation on all frames, as shown in Figure 3. It codes the I frames in isolation without any temporal prediction and the P frames using forward-motion compensation starting with the last I frame. B frame coding uses the surrounding I and/or P frames to perform forward- and backward-motion compensation. In addition, a given block in a P or B frame may be coded with or without motion compensation. The MPEG-1 standard identifies one sequence of I, B, and P frames as a group of pictures (GOP).

Motion estimation

All of the motion estimators operate only on the luminance images and compute motion vectors using block matching. With the exception of Digicipher, all motion estimators produce motion vectors that are accurate to a half pixel. Figure 4 illustrates the block-matching process. A motion vector is the spatial displacement between a block of pixels in the current frame and the most similar block in the previous frame (except for ADTV B frames). The similarity or distance measure is the mean of the absolute value of the differences between corresponding pixels in the two blocks. Other distance measures (mean square error) may provide improved performance but are much more costly to implement than the mean absolute error.

The number of block matches (and therefore the motion estimator complexity) is proportional to the search area. The DSC motion estimator has a search range of ±32 pixels, horizontally and vertically (per 1/60th second). The ADTV motion estimator has a −32/+31-pixel search range (per 1/30th second). This range applies to adjacent I and B, and P and B frames as well as to nonadjacent I and B, and I and P frames that are typically separated by two B frames (1/15th second). Motion estimation between nonadjacent frames reduces the effective search range by the number of separating frames. The horizontal search range is +31/–32 in Digicipher and +15/–16 in CCDC.

The motion estimators in both GI/MIT systems provide search ranges of +7/–8 vertically. However, the frame rate for the Digicipher system is half that of the CCDC system because Digicipher uses interlaced scanning. This effectively reduces the maximum speed of an object that can be tracked vertically by roughly a factor of two in the Digicipher system over the CCDC system and results in approximately equal horizontal tracking speeds. Similarly, the maximum tracking speed of the ADTV system is half that of the DSC system because of interlaced scanning in ADTV.

Predictive motion compensation loop

In the motion-compensated prediction loop shown in Figure 5, the motion-compensated predictor applies motion vectors to the blocks of pixels in the reconstructed frame buffer, which contains a prediction of the decoded picture. The motion-compensated predicted frame or displaced frame is subtracted from the original frame to produce the displaced frame difference. The DFD will be near zero intensity for areas with predictable motion and nonzero where detail was uncovered or motion was not just a translation in the image plane.

The DFD signal results from the removal of temporal redundancy. The DCT is applied to the DFD in preparation for the removal of spatial redundancy by coefficient quantization. A reconstructed form of the DFD is then produced by inverse quantization and inverse DCT. The system then adds the reconstructed or predicted DFD to the motion-compensated frame to produce the next reconstructed frame, which completes the coding cycle.

Decoder loop

After deformatting, inverse quantization of DCT coefficients and inverse DCT, the reconstructed DFD enters the decoding loop. See Figure 6.

The system applies the decoded motion vectors to the reconstructed frame to produce the reconstructed displaced frame. The reconstructed DFD is then added to the reconstructed displaced frame to produce the reconstructed frame, which is displayed after conversion to RGB. In the ADTV decoder, additional frame buffering is required to store extra I or P frames to reconstruct the B frames.

Prediction leak

A perfect motion-compensated prediction loop is not practical without some form of imperfection or "leak" in the prediction. This imperfection may either be constant or variable and periodic. If a sequence of input images were perfectly predictable, the decoder would not receive coefficient data and therefore could not reconstruct the picture after initialization. Decoder initialization occurs after the channel is

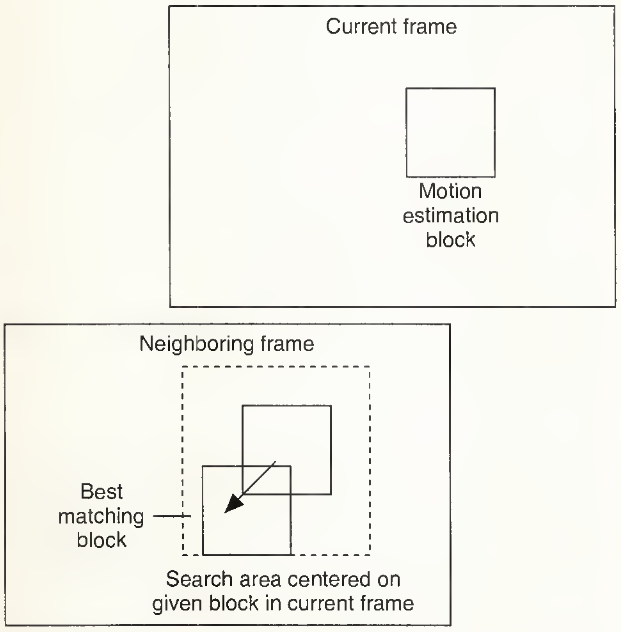


Figure 4. Block matching.

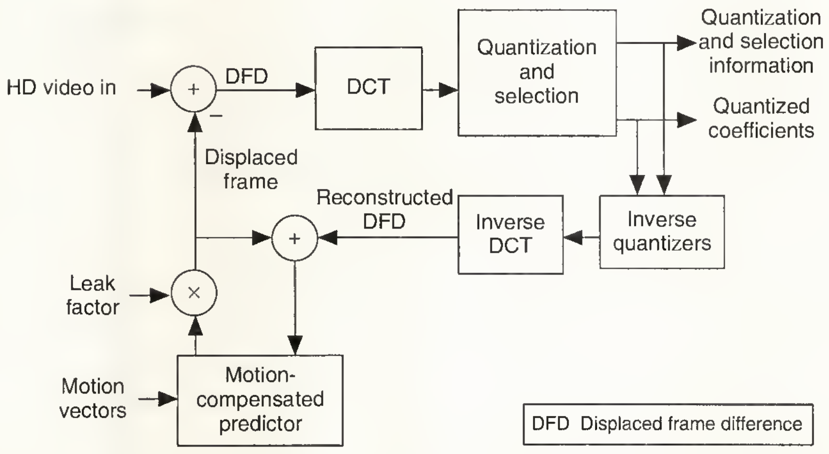


Figure 5. Motion compensation loop.

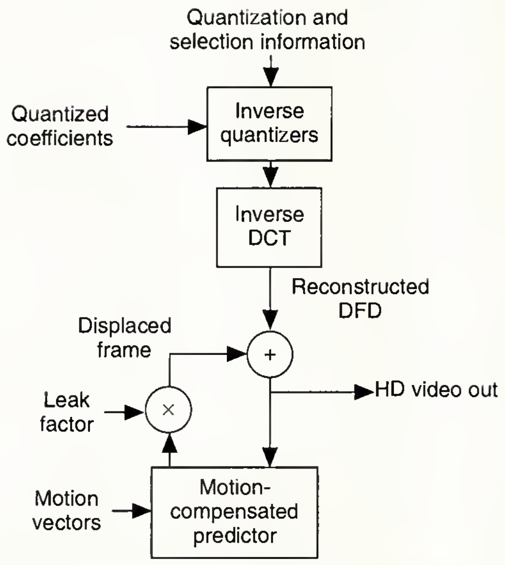


Figure 6. Decoder loop.

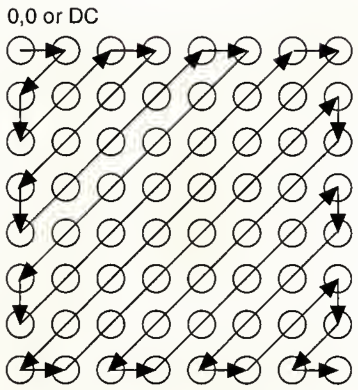


Figure 7. MPEG-1 zigzag scan of DCT coefficients.

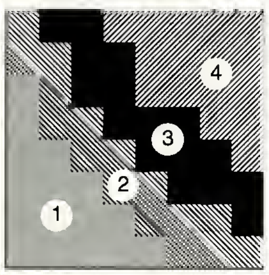


Figure 8. Four vector-coding regions used in CCDC system.

changed, or the signal is lost and then recovered.

The motion vectors and the perfect DFD for a coded still image will be zero, and the reconstructed DFD in the decoder will also be zero. If the decoder were started with a sequence of zero DFD frames, a blank or zero reconstructed frame would result. If the viewer changed to a channel transmitting a coded still image, the still image would never appear. A portion of the original picture must be mixed with the DFD in the encoder to allow the decoder to synchronize to the encoder after decoder initialization.

The use of a "leaky predictor" also improves recovery from channel impairments. The speed of recovery from channel changes and channel impairments is directly proportional to the amount of leak in the prediction. Figure 6 showed a motion-compensated prediction loop with a leak. A leak factor of 1 provides perfect prediction; a leak factor between zero and 1 mixes a portion of the original image into the DFD; and a leak factor of zero forces only intraframe coding of each frame.

Depending on the system, the leak factor may vary spatially, temporally, or both. The ADTV system satisfies the need for a leak with periodic intraframe coded I frames (a leak factor of zero). P and B frames use spatially varying leak factors of zero or 1. The GI/MIT systems force multiple columns of blocks to be intraframe coded (a leak factor of zero), while the remaining blocks can have a spatially varying leak factor of zero or 1. The columns are moved horizontally in successive frames to periodically refresh the entire picture. The DSC system does not vary the leak factor spatially but provides a leak factor that can assume one of a set of values between zero and 1 for a given frame. A leak factor close to 1 allows a usable picture to be extracted from the reconstructed DFD in the decoder to facilitate fast recovery from channel changes.

Coefficient selection and quantization

Sufficient compression cannot be achieved unless a large fraction of the DCT coefficients are dropped and therefore not selected for quantization and transmission. The unselected coefficients are assumed to have zero value in the decoder. The digital HDTV systems use several different schemes to code the coefficient selection information.

The Digicipher and ADTV systems encode the selections and runs of drops following a zigzag pattern through the array of frequency-ordered coefficients, as shown in Figure 7. The lowest frequency coefficient is known as the DC coefficient and is proportional to the mean pixel amplitude over the 8x8 block. The selected coefficient amplitudes are coded differentially to take advantage of high spatial correlation. This method of coding the coefficient selection information and selected coefficients is used in MPEG-1 and H.261.

The CCDC system subdivides the selection patterns into four regions of equal area, as shown in Figure 8. The 16-bit selection pattern for a given region is variable-length coded. The code word also indicates if all coefficients in the higher frequency regions are zero. In this case, the coding of the selection pattern is complete for the given 8×8 block. Some of the 16-bit selection patterns may occur infrequently and therefore are not variable-length coded. These patterns are sent unchanged following an escape code. Some images with high spatial frequency content could result in coded selection patterns that are at least 64 bits in length per 8×8 DCT block.

The DSC system uses vector quantization to reduce the selection information. Vector quantization reduces a large number of patterns to a small number of representative patterns and enumerates the representatives. The set of representative patterns is established before coder operation and stored in ROM in the decoder. Data compression is achieved by replacing each pattern with the index of its representative pattern. The representative for a given pattern is found by searching the set of representatives or code book for the best match. As shown in Figure 9, a given selection pattern determines which coefficients are selected for quantization and which quantizer is used for selected coefficients. The encoder chooses the selection pattern from a given block of coefficients that results in optimal quantization error and bit rate.

The ADTV and both GI/MIT systems use one uniform quantizer for coefficient quantization, while the DSC system uses multiple nonuniform quantizers. The selection patterns in the DSC system determine which quantizers are used for each selected coefficient.

Perceptual weighting

The human visual system is not uniformly sensitive to coefficient quantization error.[9] Perceptual weighting of each source of coefficient quantization error is used to increase quantization error and lower the bit rate. The amount of visible distortion resulting from quantization error for a given coefficient should depend on the coefficient number or frequency, the local brightness in the original image, the local texture in the original image, and the duration or the temporal characteristic of the error. DC coefficient error results in mean value distortion for the corresponding block of pixels, which can expose block boundaries. This distortion is much more visible than higher frequency coefficient error that appears as random noise or texture.

Displays and human vision systems exhibit nonuniform sensitivity to detail as a function of local average brightness. Loss of detail in dark areas of the picture is not as visible as it is in brighter areas. Another opportunity for bit savings occurs in textured areas of the picture where high-frequency coefficient error is much less visible than in relatively flat areas. Brightness and texture weighting require analysis of the original image since these areas may be well-predicted in the DFD. Finally, distortion is easily masked by limiting its duration to one or two frames. This effect is most profitably

0,0 or DC

①	②	③	③	Ⓓ	Ⓓ	Ⓓ	Ⓓ
②	②	③	Ⓓ	Ⓓ	Ⓓ	Ⓓ	Ⓓ
②	③	③	Ⓓ	Ⓓ	Ⓓ	Ⓓ	Ⓓ
③	③	③	Ⓓ	Ⓓ	Ⓓ	Ⓓ	Ⓓ
③	③	③	Ⓓ	Ⓓ	Ⓓ	Ⓓ	Ⓓ
③	③	Ⓓ	Ⓓ	Ⓓ	Ⓓ	Ⓓ	Ⓓ
③	Ⓓ	Ⓓ	Ⓓ	Ⓓ	Ⓓ	Ⓓ	Ⓓ
Ⓓ	Ⓓ	Ⓓ	Ⓓ	Ⓓ	Ⓓ	Ⓓ	Ⓓ

Figure 9. Quantizer selection pattern. D = drop; 1,2, and 3 = quantizer numbers for DSC system.

used after scene changes in which the first frame or two can be severely distorted without perceptible distortion at normal speed.

Variable-length coding

The data representing information such as motion vectors, quantizer selection patterns, and transform coefficients are seldom statistically uniform. Usually, the data are statistically clustered, and the probability distributions can be estimated from analysis of real scenes. The use of variable-length codes[10] (VLCs) takes advantage of this statistical nonuniformity by assigning short code words to the most frequent values and assigning longer words to less frequent values.

Prior to VLC code book design, the frequency statistics for each type of value (motion vectors, coefficients) are gathered, using a variety of scene material. If there is not a proper balance or variety of scenes, subsequent variable-length coding gain may be very high for some types of scenes and very low (or negative) for other types of scenes that were not represented during code book training.

The value statistics are mapped into code words such that the length of each code word is inversely proportional to the frequency of occurrence of its associated value. As shown in Figure 10 (on the next page), a set of code words results from concatenating bits from the root of the tree to a given terminal node or leaf. The tree is pruned to provide code word lengths that match the value frequencies. For example, if the most frequent value occurs at least half of the time, a 1-bit code word should be used for that value.

Variable-length encoding consists of looking up a given VLC in a table and using the value as the table index. Tracing each bit through the tree until a terminal node is reached decodes a given code word. The decoded value is stored with the terminal node.

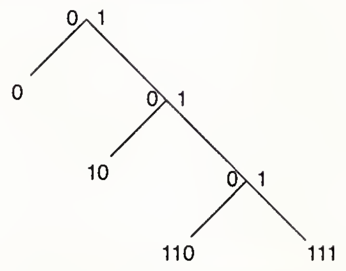


Figure 10. VLC decoding tree.

Buffer control

Motion compensation, adaptive quantization, and variable-length coding produce highly variable amounts of compressed video data as a function of time. For example, the compressed bit rate after a scene change can be several times greater than the bit rate in the channel. Therefore, efficient channel utilization requires compressed video data buffering in the encoder and decoder.

Buffer size is constrained by the maximum tolerable delay through the system and by cost. Adjusting the amount of distortion or quantization error in each image controls the fullness of the buffer. The encoder's buffer will fill more quickly when the distortion is low. A feedback control system is required to regulate the distortion level, which controls the buffer fullness to prevent overflow. Three conditions complicate the design of this control system:

- delay between a change in the distortion level and the subsequent change in buffer fullness;
- perceptual constraints on the instantaneous and average distortion levels; and
- difficulties in modeling the bit rate as a function of distortion level.

We can minimize the visibility of distortion due to an increase in scene complexity by smoothly increasing the distortion level. This is facilitated by accurately modeling the rate-versus-distortion level because an accurate model allows the desired buffer fullness to be achieved for each frame.

Compressed video formats

The compressed video data must be organized into a format that can be reliably transmitted through an imperfect channel and allow the decoder to recover quickly from the loss of data in the channel. In addition, storage and manipulation of the compressed video and transmission over alternate media (packet networks) motivate the inclusion of block, frame, and program type descriptors in the format.

Robust transmission is primarily achieved by dividing the compressed video bit stream into fixed-length blocks and applying error correction processing to each block. Error correction overhead or parity bits are added to each block before transmission. The receiver then uses the parity bits to attempt correction of transmission errors. Uncorrected errors are detected in the receiver, and blocks containing them are not sent to the decoder. If a block is lost, the system conceals the error in the decoder to minimize its visual impact.

The amount of picture that is degraded by the loss of a block of compressed video data depends on the format. Since most of the data consists of concatenated variable-length code words, VLC restart pointers must occur at regular intervals in the bit stream. More frequent placement of these pointers will reduce the amount of picture degradation but will also consume more of the bit rate.

A problem associated with the terrestrial broadcast of modulated digital data is the sudden loss of received data when the noise threshold has been exceeded.[11] This "cliff effect" occurs near the fringe of the broadcast coverage area or wherever the error correction system is overcome by noise or interference in the signal. The DSC and ADTV systems include schemes that avoid the cliff effect by transmitting the most important data using a more robust mode.

The ADTV system divides the terrestrial broadcast channel into two separate channels. One channel uses higher power than the other and is positioned to minimize interference with an NTSC cochannel. It transmits a fixed fraction of the most important data using the higher power channel, which is more robust than the other channel. The majority of the remaining data is transmitted in the wider and lower power channel. Complete loss of picture is avoided if the higher priority data is correctly received.

The DSC system performs time division multiplexing between two modulation modes. One mode provides much more immunity to channel impairments than the other mode but yields half the data rate of the less robust mode. Some or all of the compressed video data are chosen for more robust transmission depending on scene complexity and desired picture quality level. The encoder selects the transmission mode for each error correction block of data and chooses for more robust transmission those blocks containing more important data. Most scenes can be transmitted entirely in the more robust mode.

Techniques for efficient storage and manipulation of the compressed video (and audio) data will be necessary to provide consumers with VCR capabilities such as speed search and freeze frame. Since all of the decoders contain reconstructed frame storage, freeze frame is simply provided. Each system provides speed search with varying degrees of efficiency and picture quality. This function is provided by subsampling the blocks of compressed video data from tape that is moving at higher than normal speed across the head. The block descriptors are then used to partially reconstruct

The compressed bit rate after a scene change can be several times greater than the bit rate in the channel.

the video. The ability to decode randomly selected portions of compressed video data also facilitates multimedia computing applications where multiple received and stored video data streams could be manipulated quickly within a window system.

THE IMPLICATIONS OF DIGITAL VIDEO CODING and transmission are very positive and far reaching. Applications to multimedia computing, interactive television, video-archiving systems, and video telephony will lead to a convergence of entertainment, computing, and communications that will greatly enrich the human experience.

The digital video revolution also has positive implications for international standards. Conversion between scanning formats is much easier in the digital domain, and translation between video coding standards is greatly facilitated by digital representation. This may lead to an eventual worldwide unification of video standards.

References

1. *DigiCipher HDTV System Description*, GI Corporation, San Diego, Calif., Aug. 1991.
2. *Channel Compatible Digicipher HDTV System*, American Television Alliance, Massachusetts Institute of Technology, Cambridge, Mass., Apr. 1992.
3. *Digital Spectrum Compatible HDTV System*, Zenith Electronics Corporation, Glenview, Ill., Nov. 1991.
4. *Advanced Digital Television System Description*, Advanced Television Research Consortium, David Sarnoff Research Center, Princeton, N.J., and Philips Laboratories, Briarcliff Manor, N.Y., Jan. 20, 1992.
5. "ISO CD 11172-2: Coding of Moving Pictures and Associated Audio for Digital Storage Media at Up to About 1.5 Mbit/s," International Standards Organization, Geneva, Nov. 1991.
6. A. Netravali and B. Haskell, *Digital Pictures: Representation and Compression*, Plenum Press, New York, 1988.
7. J. Amanatides and D. Mitchell, "Antialiasing of Interlaced Video Animation," *Computer Graphics*, Vol. 24, No. 4, Aug. 1990, pp. 77-85.
8. K.R. Rao, "Discrete Cosine Transform," Academic Press, New York, 1990.
9. A. Netravali and B. Prasada: "Adaptive Quantization of Picture Signals Using Spatial Masking," *IEEE Proc.*, Vol. 65, No. 4, Apr. 1977, pp. 536-548.
10. D.A. Huffman, "A Method for the Construction of Minimum Redundancy Codes," *Proc. IRE*, Vol. 40, No. 1089, 1952.
11. K. Feher, *Advanced Digital Communications*, Prentice-Hall, Inc., Englewood Cliffs, N.J., 1987.

The author's biography and picture appear on p. 12 in this issue.

Direct any comments regarding this article to Eric Petajan, AT&T Bell Laboratories, 600 Mountain Avenue, Murray Hill, NJ 07974; or e-mail at edp@allegra.attmail.com.

Reader Interest Survey

Indicate your interest in this article by circling the appropriate number on the Reader Service Card.

Low 153 Medium 154 High 155

Architecture and Implementation of ICs for a **DSC-HDTV Video Decoder System**

Zenith and AT&T developed the Digital Spectrum Compatible high-definition television (HDTV) system for evaluation by the US FCC Advanced Television Test Center. The system, specifically designed to minimize consumer receiver cost, uses a video decoder subsystem to decode the digital video bit stream and reconstruct video fields. Two new VLSI devices implement this video decoder in a cost-effective manner. We comment on the architecture and hardware complexity of these devices.

Obed Duardo

Scott C. Knauer

John N. Mailhot

Kalyan Mondal

Tommy C. Poon

AT&T Bell Laboratories

Zenith and AT&T's all-digital high-definition television simulcast system called Digital Spectrum Compatible[1] combines powerful video compression technology and a unique simulcast transmission system. The DSC-HDTV compression technology handles high-detail, complex motion video and adapts to scene changes and receiver channel changes quickly. The companies specifically designed the system to optimize the cost of the receiver, considering both processing and memory requirements.

Figures 1 and 2 contain block diagrams of the DSC-HDTV transmitter and receiver. The transmitter video encoder compresses the HDTV images by removing temporal, spatial, and amplitude redundancies. Figure 3 shows an overview of the video encoder. At the receiver, the compressed video data enters the video decoder subsystem, which reconstructs the coded HDTV image. Figure 4 shows a block diagram of the DSC-HDTV video decoder subsystem. The functions of the decoder subsystem have been partitioned into two unique VLSI devices: the deformatter IC and the motion compensator/inverse discrete cosine transform (MC/IDCT) IC. A few commercial DRAM devices complete the subsystem. The video encoder also uses the MC/IDCT.

The motion estimator video encoder detects motion of blocks from frame to frame to generate motion vectors, which the MC/IDCT uses to predict the current frame from the previous frame. The predicted frame is subtracted from the actual frame to generate the displaced frame difference (DFD). By scaling the prediction prior to the subtraction, the DFD contains a fraction of the original frame. A two-dimensional discrete cosine transform (DCT) removes spatial redundancy in the DFD. Quantization of the DCT coefficients is based on perceptual criteria as computed by the forward analyzer, which uses a model of the human visual system to minimize perception of coding artifacts. The buffer/formatter packs the motion vectors, compressed DCT coefficients, and the coding parameters into a compressed format. Each HDTV video frame of 1,280×720 pels is compressed from 994 Mbits per second to between 9 and 17 Mbps and can be transmitted on a 6-MHz channel.

Deformatter

The deformatter IC converts the formatted data back to motion vectors, DCT coefficients, and coding parameters. The AT&T 0.5-micron CMOS deformatter occupies an area of 60 sq mm. Operating at 37.5 MHz, the deformatter dissipates an estimated 2 watts of power and processes both luminance and chrominance values.

Figure 5 shows a block diagram of the deformatter. The channel data parser determines

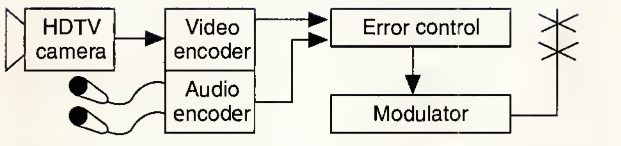


Figure 1. DSC-HDTV transmitter.

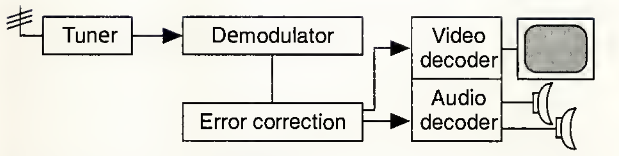


Figure 2. Receiver.

slice boundaries from markings in the data; the slice data is tagged for distribution to one of the variable-length decoding (VLD) engines or to the global data controller. All data is buffered in the compressed channel buffer after tagging. During each video field time, the VLD engines and global data controller must decode all of the compressed data for the field. Rather than amortize the decoding time required by more complicated sections of the image over the other parts, potentially requiring large amounts of buffering of uncompressed data, our implementation always buffers in the compressed domain and decodes data at the worst case burst rate.

In the DSC-HDTV system, each 1,280×720 video frame consists of 300 slices of 64×48 pels. Each slice consists of 48 luma transform blocks (8×8 pels each) and 24 chroma transform blocks (8×8 samples each); potentially, 4,608 coefficient VLC code words could be decoded in each slice. During each slice time 3,072 cycles of the 75-MHz sampling clock, or 1,536 cycles of the 37.5-MHz processing clock, take place. Three VLD engines capable of decoding one code word per cycle are needed to maintain the worst case burst rate of coefficient data. Figure 6 on the next page shows such an engine.

In this figure, the buffer address controller requests data whenever the burst FIFO is not full. This FIFO is sized as a function of the worst case compressed slice length. The feeder subfunction provides LSB-justified code words to the length detector; the length output feeds back into the feeder to align the next code word. The critical path of this engine is the length detection op-

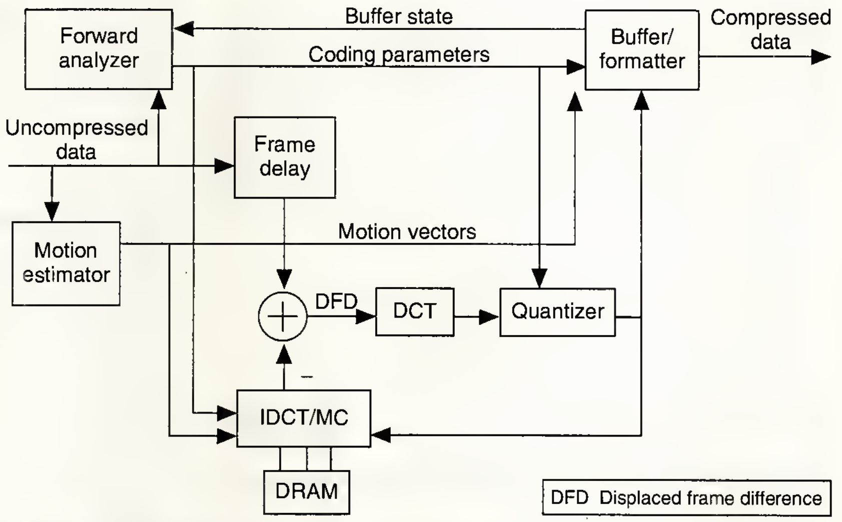


Figure 3. Video encoder overview.

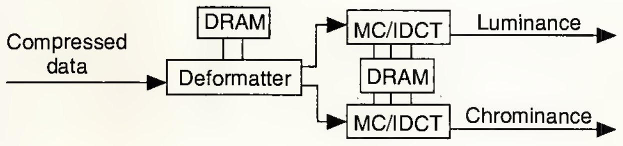


Figure 4. Video decoder overview.

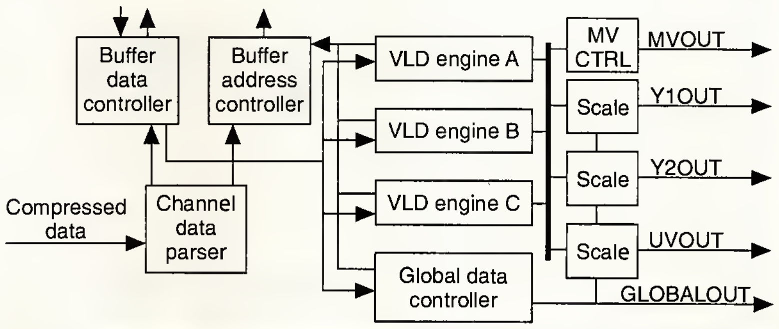


Figure 5. Deformatter IC.

eration feeding back into the feeder, which must be a one-cycle operation for a 1-code-word/cycle decoding engine.

Once the code words have been separated and the lengths are known, the decoding engine needs only to decode the code words back into their original values. The fixed decoder subfunction accomplishes this operation.

The slice sequence controller subfunction is a state machine that sequences through the data elements in a slice. The DSC-HDTV compressed format uses a number of different variable-length codes to encode motion information, selection information, and coefficient data. The slice sequence controller feeds a sequence of code identifiers to the length detector and fixed decoder. The controller receives as feedback the coefficient selection information, which identifies the proper variable-length codes for the coefficient data.

Because the slice decoding takes place in three parallel engines, a minimal amount of reconstruction buffering is needed to put the data back into the proper sequence for the MC/IDCT chip. Through the use of fixed time-slot scheduling of the reconstruction bus, this amount of storage is limited to one uncompressed slice per engine.

Global data is buffered in the compressed channel buffer along with the local data to which it corresponds. The global data controller subfunction decodes the global data and passes it to the other parts of the decoder. The largest item in the global data is the set of subband scaling factors; these are decoded and fed to the scaling hardware. Each scaling engine is realized as a set of shift-and-add constant multipliers in which the coded scale factor selects the adds to enable. The scaled value is clamped to the valid precision range.

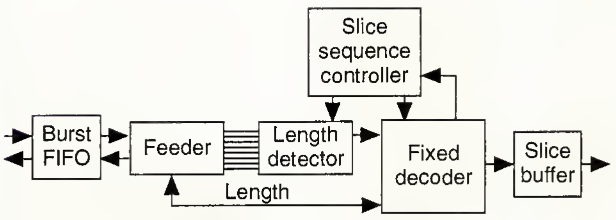


Figure 6. VLD engine.

Motion compensator and IDCT

The HDTV motion compensator and IDCT chip (MC/IDCT IC) reconstructs frames from the deformatter-decoded coefficients. The 164-pin MC/IDCT maintains a 2-pel bus width and operates at 37.5 MHz to achieve the desired throughput. The AT&T 0.5-micron CMOS device occupies an estimated area of 56 sq mm and dissipates an estimated 2 watts of power. The external frame buffer requires a 256K×32 DRAM for luminance and a 256K×16 DRAM for chrominance. Figure 7 shows the architecture.

The IDCT converts transmitted coefficients to the displaced frame difference, the difference between an original frame and a predicted frame. The previous frame stored in external memory generates each predicted frame by applying the transmitted motion vectors to blocks as small as 8×8 pels. The MC/IDCT can handle subpixel motion vectors in the pel interpolator by calculating fractional motion pel values from adjacent previous-frame pels. The algorithm scales the resultant frame by the DF factor to generate the displaced frame. By allowing the DFD to contain a fraction of the original image, the DF factor facilitates fast recovery from channel errors and channel changes. The DFD and DF are added and temporal filtering[2] is applied to reconstruct the image.

An example of pel interpolation is given in Figure 8. The motion vector for this example is one-half pel in the *x* direction and one-half pel in the *y* direction. The dark circles represent the previous-frame pel data from the external-frame buffer. The white circles represent pels for the current frame, which are calculated from the previous-frame pels. In this example $A(n)$ equals the average of $A(n-1)$, $B(n-1)$, $C(n-1)$, and $D(n-1)$ in the previous frame.

The temporal filter outputs a weighted aver-

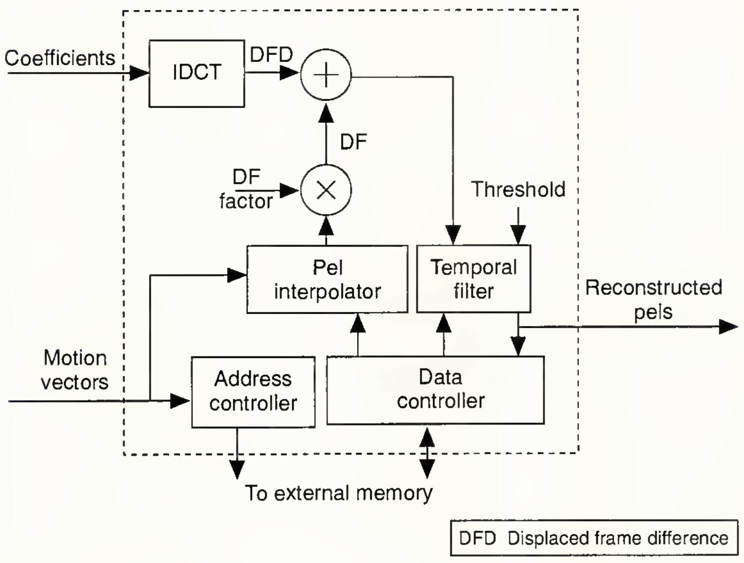


Figure 7. Motion compensator/IDCT IC.

age of the current frame pel and the previous frame pel whenever the difference of the two pels is less than the transmitted threshold. If the difference is greater than the threshold, the current frame value is passed. This filtering reduces the level of noise in stationary areas and helps to eliminate the propagation of coding errors.

The IDCT function receives decoded transform coefficients of Y, U, and V and inversely transforms them to obtain 8×8 blocks of pels. The 2D IDCT can be performed by a concatenation of two 1D IDCTs.

$$[x]^T = [C]^T\left[[C]^T[X]\right]^T$$

Figure 9 displays the overall architecture of an 8×8 2D IDCT module. In keeping with the output format of the luminance from the deformatter chip, two coefficients are received in each 37.5-MHz cycle; the 1D transform processors and transform buffer are optimized around this twofold parallelism.

Two techniques simplify the 8-point IDCT operation. First, the transform operation breaks down into two parts of odd and even coefficients. By permuting the order of coefficients and pels, the transform matrix is arranged so that elements on full or partial diagonals have the same absolute values (Figure 10). This leads to the hardware architecture illustrated in Figure 11 on the next page. In the odd (even) coefficient processor, the coefficients are simultaneously multiplied by four (three) constants, and the results feed first to sign change units and then to shared accumulators. Every four cycles, the results in the accumulators feed into a parallel-in/serial-out shift register and shift out. Addition and subtraction of these outputs complete the calculation. Memory addressing where possible, and simple arrangements of two input multiplexers and registers elsewhere, change the pel and coefficient order. Note that this architecture produces two results each clock cycle.

The second technique that simplifies the transform exploits the fact that the architecture multiplies the same coefficient by several constants at once. It is possible to multiply the coefficients by small factors to produce intermediate results that can be shared in the computation of the multiple products. Figure 12 illustrates this technique. The intermediate results are shifted and added to provide 16-bit precision multiplications. The particular numbers shown are only one set of many that will solve the problem and merely represent an illustration

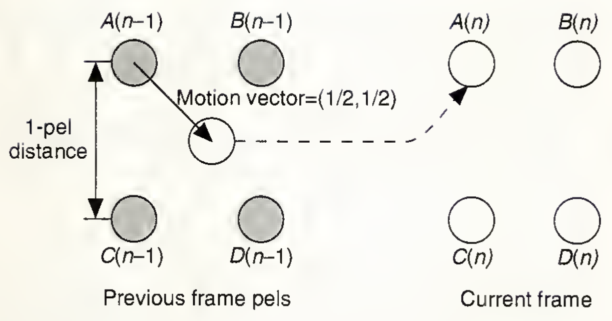


Figure 8. Pel interpolator example.

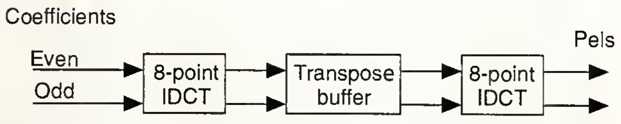
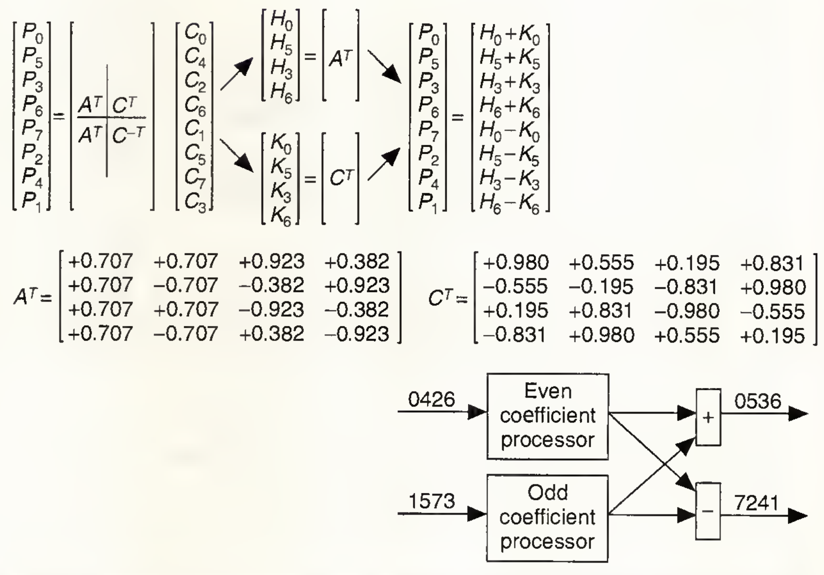


Figure 9. The 8×8 IDCT architecture.



$$\begin{bmatrix}P_0\\P_5\\P_3\\P_6\\P_7\\P_2\\P_4\\P_1\end{bmatrix}=\begin{bmatrix}A^T & C^T\\ A^T & C^{-T}\end{bmatrix}\begin{bmatrix}C_0\\C_4\\C_2\\C_6\\C_1\\C_5\\C_7\\C_3\end{bmatrix} \qquad \begin{bmatrix}H_0\\H_5\\H_3\\H_6\end{bmatrix}=\begin{bmatrix}A^T\end{bmatrix} \qquad \begin{bmatrix}K_0\\K_5\\K_3\\K_6\end{bmatrix}=\begin{bmatrix}C^T\end{bmatrix} \qquad \begin{bmatrix}P_0\\P_5\\P_3\\P_6\\P_7\\P_2\\P_4\\P_1\end{bmatrix}=\begin{bmatrix}H_0+K_0\\H_5+K_5\\H_3+K_3\\H_6+K_6\\H_0-K_0\\H_5-K_5\\H_3-K_3\\H_6-K_6\end{bmatrix}$$

$$A^T=\begin{bmatrix}+0.707 & +0.707 & +0.923 & +0.382\\ +0.707 & -0.707 & -0.382 & +0.923\\ +0.707 & +0.707 & -0.923 & -0.382\\ +0.707 & -0.707 & +0.382 & -0.923\end{bmatrix} \qquad C^T=\begin{bmatrix}+0.980 & +0.555 & +0.195 & +0.831\\ -0.555 & -0.195 & -0.831 & +0.980\\ +0.195 & +0.831 & -0.980 & -0.555\\ -0.831 & +0.980 & +0.555 & +0.195\end{bmatrix}$$

Figure 10. IDCT even/odd simplification.

of the general technique. Pipeline registers are not shown; these and half-adder counts, fan-out, and layout topology are the parameters for choosing the exact values of the intermediate factors.

The same multiply/accumulate blocks support the forward DCT, a transform architecture that can also be applied to 16 element transforms, sine transforms, overlapped transforms, and nonpower of two transforms. Since only a few stages of addition are used, less error accumulates in the use of intermediate results. This process easily complies with CCITT requirements for IDCT precision.

An IDCT block based on this architecture can be implemented very efficiently using full-custom techniques; the adders and data paths are realized as pitchmatched arrays with buffers pitchmatched to the adder and other cells. Even in current 1.0-micron CMOS technology, this novel architecture permits implementation with at least a 30 percent smaller area than commercially available devices.[5]

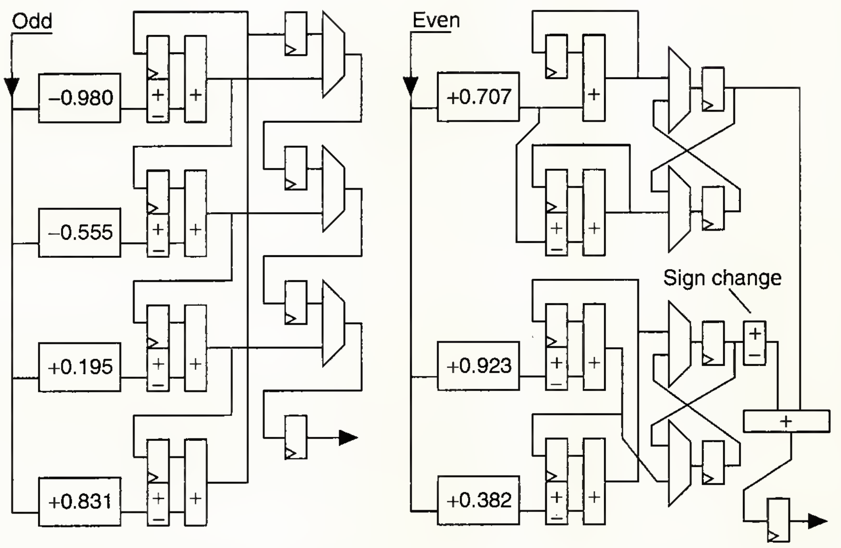


Figure 11. IDCT coefficient processors.

WE DISCUSSED THE ARCHITECTURE and implementation of the video decoder function of the DSC-HDTV receiver. The level of functional integration of these ICs is substantial; however, next-generation 0.5-micron CMOS technology will produce an ideal low-cost implementation suitable for consumer television use.

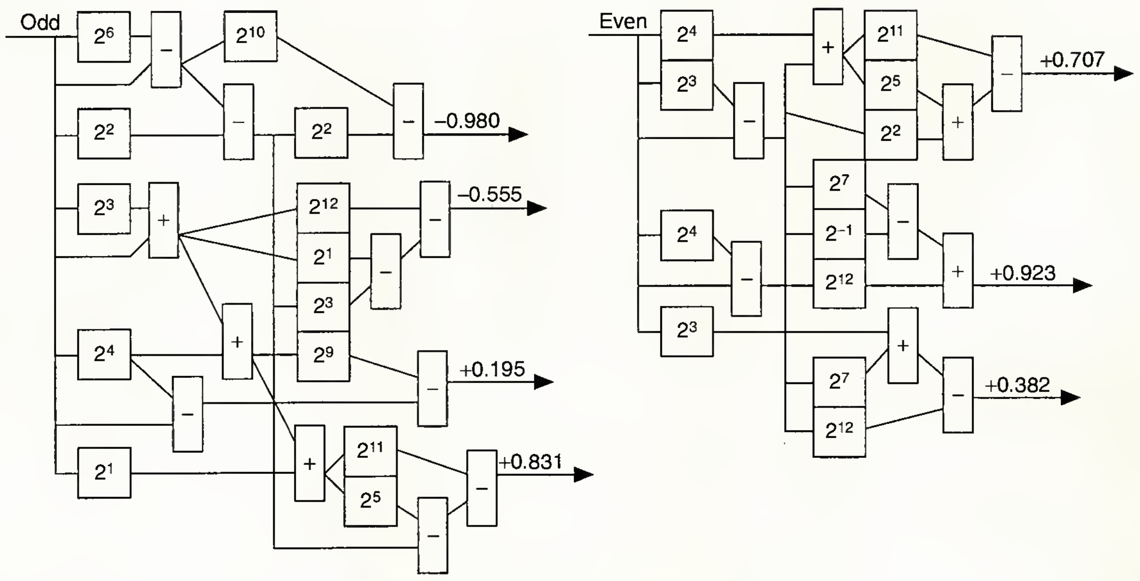
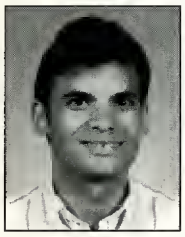


Figure 12. IDCT constant multipliers.

References

1. "Zenith-AT&T Digital Spectrum Compatible Technical Details," Zenith and AT&T, Sept. 23, 1991.
2. A.N. Netravali and B.G. Haskell, *Digital Pictures Representation and Compression,* Plenum Press, New York, 1988.
3. T. Liu and W.F. Wedam, "Hardware Implementation and Cost of Decoders for Digital HDTV," *Proc. ICCE,* 1991.
4. S.M. Lei and M.T. Sun, "An Entropy Coding System for Digital HDTV Applications," *IEEE Trans. Circuits and Systems for Video Technology,* Vol. 1, No. 1, Mar. 1991, pp. 147-154.
5. P.A. Ruetz et al., "A High-Performance Full-Motion Video Compression Chip Set," *IEEE Trans. Circuits and Systems for Video Technology,* Vol. 2, No. 2, pp. 111-122, June 1992.



Obed Duardo is a member of the technical staff at AT&T Bell Laboratories in Murray Hill, New Jersey, where he works on board- and IC-level designs for HDTV, NTSC, and PAL video compression. His technical interests include HDTV, VLSI, and circuit pack architecture design, encryption, error detection and correction, and RISC processors.

Duardo received his BE and ME degrees from Stevens Institute of Technology in Hoboken, New Jersey. He is a member of Tau Beta Pi.

Scott C. Knauer heads the Digital Architectures Research Department in the Computing Systems Research Laboratory at AT&T Bell Laboratories. He is part of the AT&T/Zenith team developing an HDTV standard (including a hardware prototype transmitter and receiver) for the FCC competition and continues his activities in broadband (2.4-Gbit/line) switching projects. He and Alan Huang developed the self-routing, nonblocking Batcher-banyan fabric and several systems based upon variations of this principle.

Knauer received the BSEE from Rutgers and the MSEE and PhD in electrical engineering from Stanford University. He is a member of the IEEE and the Computer Society.

John N. Mailhot of AT&T Bell Laboratories is engaged in system architecture and integration of the AT&T/Zenith digital HDTV system. He received a BS degree in electrical engineering and a BS degree in computer science from the Massachusetts Institute of Technology. He is a member of Eta Kappa Nu.

Kalyan Mondal, a distinguished member of the technical staff at AT&T Bell Laboratories, works in the areas of video compression VLSI design, application-specific signal processor design, CAD for DSP design, and general-purpose DSP VLSI design. Previously, he held academic positions at UC Santa Barbara and Lehigh University in Bethlehem, Pennsylvania.

Mondal holds a PhD in electrical engineering from the University of California at Santa Barbara. He is a senior member of the IEEE and a member of the Computer Society, the Board of Governors of the IEEE Circuits and Systems Society, Eta Kappa Nu, and Sigma Xi. He is a cowinner of the 1985 IEEE International Solid-State Circuits Conference Outstanding Paper, "A Programmable Digital Signal Processor with 32b Floating-Point Arithmetic."

Tommy C. Poon is the supervisor of a group at AT&T Bell Laboratories involved in NTSC, HDTV, and ATM. His technical interests include VLSI design and fabrication, digital data processing, and data transportation.

Poon received his BSEE and MSEE from Rutgers University and his PhD in electrical engineering from Columbia University. He is a member of the IEEE.

Direct questions about this article to Obed Duardo, AT&T Bell Laboratories, 600 Mountain Avenue, Murray Hill, NJ 07974; or e-mail at od@allegra.att.com.

Reader Interest Survey

Indicate your interest in this article by circling the appropriate number on the Reader Service Card.

Low 156 Medium 157 High 158

A 160-Mpixel/s IDCT Processor for HDTV

A fabricated, tested, fully functional 40-MHz device performs the 8×8 inverse discrete cosine transform for digital HDTV decoders. It converts four 14-bit DCT coefficients into four 11-bit pixel values each cycle. Fixed-coefficient multiplier Wallace trees (in which partial products are rounded before summation) help compute the inner products. The 31,000-gate device was implemented in a 10.5-mm die using a 1-micron CMOS array-based process.

Peter A. Ruetz

Po Tong

LSI Logic Corp.

The IDCT is the core of video compression decoders proposed for video conferencing, image archiving, multimedia, and HDTV purposes. For HDTV, the decoder is of greater interest than the encoder because more decoders than encoders will be used. HDTV also presents far more difficult technical challenges than other applications because of the proposed over-100-Mpixel/s rates (104 Mpixels/s in Lei and Sun[1] and 130 Mpixels/s in Barbero, Cucchi, and Stroppiana.[2] In addition, the design of the IDCT is more difficult than the DCT because of its more stringent computational accuracy requirements.

The literature reports various other DCT/IDCT chips. Gottlieb, Sun, and Chen[3] discuss a 73,000-transistor, 16×16 DCT chip that uses distributed arithmetic. It processes 14.3 Mpixels/s in a 2-micron CMOS technology. A 34,000-transistor, 8×8 DCT chip reported by Vitterli and Ligtenberg[4] uses a fast algorithm with only 12 multiplications and processes 10 Mpixels/s in a 2.5-micron technology. Chen, Smith, and Fralich[5] used an approach similar to Gottlieb, Sun, and Chen and an 0.8-micron BiCMOS process for a 100-Mpixel/s chip.

Here, we present an 8×8 IDCT chip for digital HDTV that converts four 14-bit DCT coefficients into four 11-bit pixel values each clock cycle. Fabricated using a 1-micron CMOS array-based process, this chip is fully functional at 40 MHz, giving it a throughput of 160 Mpixels/s. Computing the IDCT inner products with fixed-coefficient multiplier Wallace trees produces this high throughput. To minimize hardware, the partial products in the Wallace trees are rounded before summation. Careful rounding maintains a high accuracy of computation for keeping the encoder and decoder predictors from mistracking.

Architecture

The two-dimensional 8×8 inverse discrete cosine transform (IDCT) is defined as

$$P=(1/4)CD^tC^t,$$

where D is an 8×8 matrix of DCT coefficients, P is an 8×8 matrix of pixel values, and C is the 8×8 transform matrix given by

$$C_{i,j}=\cos(2i+1)j\pi/16 \text{ for } j\neq 0, \text{ and } C_{i,0}=1/\sqrt{2}.$$

We can rewrite the 2D transform $P=(1/4)CD^tC^t$ as

An earlier version of this article appeared in the *Proceedings of the IEEE 1992 Custom Integrated Circuits Conference,* pp. 26.4.1-26.4.4.

0272-1732/92/1000-0028$03.00 © 1992 IEEE

$$P = (1/2)C[(1/2)CD]^t,$$

and then decompose it into two stages of 1D IDCTs of the form

$$Y = (1/2)CX,$$

where X and Y are 8-point column vectors. Each stage performs eight 1D IDCTs, and the two stages are separated by an 8×8 matrix transposition.

Figure 1 shows the overall architecture of the chip. The core of the circuit is the 1D IDCT processor, which performs the 8×8 matrix multiplication $Y = (1/2)CX$ in one cycle. In other words, an 8-point vector is processed every cycle, during which eight 8-point inner products are performed. This time-multiplexed unit operates on first-stage and second-stage data in alternate cycles. The transposition circuit converts the incoming row-organized data in the first stage into column-organized data in the second stage and occupies one third of the die area due to the high I/O bandwidth. The input and output ports pass four values of the 8-point vector each cycle.

By using only one stage of the fast transform decomposition,[6] we can write the 8×8 matrix C in the 1D IDCT $Y = (1/2)CX$ as the product AB, where

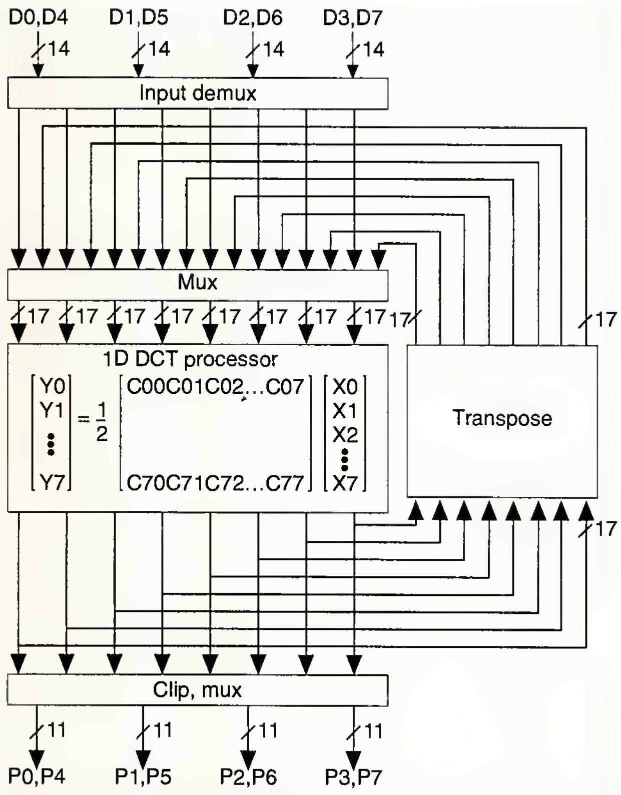


Figure 1. The 8×8 IDCT architecture.

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & -1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix}$$

$$B = \begin{bmatrix} \frac{1}{\sqrt{2}} & 0 & \cos\frac{2\pi}{16} & 0 & \frac{1}{\sqrt{2}} & 0 & \cos\frac{6\pi}{16} & 0 \\ \frac{1}{\sqrt{2}} & 0 & \cos\frac{6\pi}{16} & 0 & \frac{-1}{\sqrt{2}} & 0 & \cos\frac{18\pi}{16} & 0 \\ \frac{1}{\sqrt{2}} & 0 & -\cos\frac{6\pi}{16} & 0 & \frac{-1}{\sqrt{2}} & 0 & -\cos\frac{18\pi}{16} & 0 \\ \frac{1}{\sqrt{2}} & 0 & -\cos\frac{6\pi}{16} & 0 & \frac{1}{\sqrt{2}} & 0 & -\cos\frac{6\pi}{16} & 0 \\ 0 & \cos\frac{7\pi}{16} & 0 & \cos\frac{21\pi}{16} & 0 & \cos\frac{35\pi}{16} & 0 & \cos\frac{49\pi}{16} \\ 0 & \cos\frac{5\pi}{16} & 0 & \cos\frac{15\pi}{16} & 0 & \cos\frac{25\pi}{16} & 0 & \cos\frac{35\pi}{16} \\ 0 & \cos\frac{3\pi}{16} & 0 & \cos\frac{9\pi}{16} & 0 & \cos\frac{15\pi}{16} & 0 & \cos\frac{21\pi}{16} \\ 0 & \cos\frac{\pi}{16} & 0 & \cos\frac{3\pi}{16} & 0 & \cos\frac{5\pi}{16} & 0 & \cos\frac{7\pi}{16} \end{bmatrix}$$

Together with scaling the matrix coefficients by $1/\sqrt{2}$, this process reduces the number of unique nontrivial multiplications from 64 to 20. This compares favorably with many fast transforms,[4,6] which require only 12 to 16 multiplications. The fast transforms, however, suffer greatly by the need to form true binary results at the input to each multiplier, increasing the number of carry-propagate adders. This in turn requires the use of more pipeline stages to maintain performance.

Figure 2 on the next page shows the 1D IDCT processor, which has two pipeline stages. Each Wallace tree contains one pipeline stage and computes the sum of up to four fixed-coefficient products. In the second pipeline stage, the binary result of each Wallace tree out-

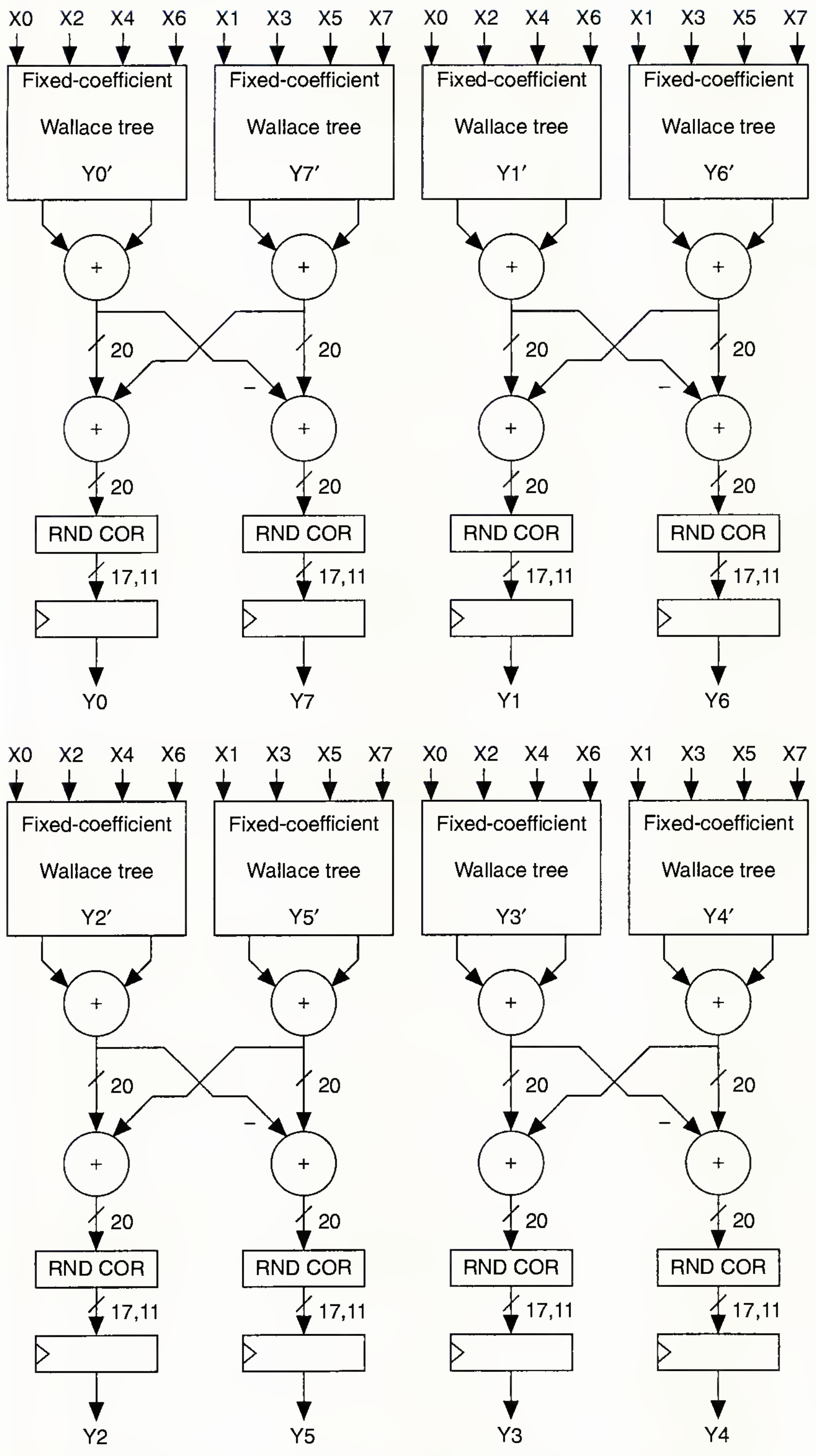

X0 X2 X4 X6
X1 X3 X5 X7
Fixed-coefficient Wallace tree Y0′
Fixed-coefficient Wallace tree Y7′
Fixed-coefficient Wallace tree Y1′
Fixed-coefficient Wallace tree Y6′
Fixed-coefficient Wallace tree Y2′
Fixed-coefficient Wallace tree Y5′
Fixed-coefficient Wallace tree Y3′
Fixed-coefficient Wallace tree Y4′
+
−
20
RND COR
17,11
Y0
Y7
Y1
Y6
Y2
Y5
Y3
Y4

Figure 2. ID 1DCT processor. (RND COR indicates rounding correction.)

put forms, and the final addition and subtraction to get the output values takes place. Optimizing the second carry select adder (CSA)—given the skewed bit arrival times caused by the first CSA—incurs only 2 nanoseconds of additional delay.

Computation accuracy

Accurate computation is extremely important in keeping the encoder and decoder predictors from mistracking. We used the CCITT H.261 video telephony error specifications,[7] scaled by making it more strict by a factor of four. This current chip is designed for input and output data with two more bits of precision than required by CCITT H.261.

We chose the transform matrix precision (16 bits) and the row result precision (17 bits) to meet the mean-square-error (MSE) requirement with full-precision partial products in the Wallace trees. We minimized hardware by truncating the partial products in the Wallace trees and rounding before summation (see Figure 3).

This technique is appropriate because in this case the multiplier inputs (the transform matrix coefficients, the DCT coefficients, and the row results) are inherently inaccurate. We used a biased scheme in rounding the partial products. The most significant discarded bit is added to the least significant bit of the truncated result. This scheme is biased in the sense that when the original number is exactly half way in between the two nearest possible results, the larger value is always chosen. To determine the number of bits that could be discarded, we investigated the various effects of partial product rounding in detail.

Figure 4 shows the effects of partial product rounding on MSE and gate count. Up to 12 bits can be discarded without exceeding the maximum allowable value of MSE (dotted line) while reducing the gate count by approximately 25 percent.

We show the overall and peak average errors in Figures 5 and 6. In each of these figures, two average error curves show biased and unbiased rounding of the 1D IDCT output. We implemented unbiased rounding at the 1D IDCT output, making it possible to discard 12 bits without exceeding the error specifications. With biased rounding, only 11 bits can be discarded. By first performing a biased round and then correcting the result, we implemented the unbiased rounding without the need for additional incrementers or adders. The correction is done by forcing the least significant bit of the result to zero if the original number falls exactly half way in between the two possible results. In this way, ties are resolved by rounding up and down with equal chances, making it unbiased. This correction circuit added only 10 gates per output with no impact on the critical path while saving 550 gates from the Wallace trees by discarding one more bit.

The architecture can be appropriately generalized to compute the 16×16 IDCT. In this case, each of the Wallace trees in Figure 2 computes the sum of up to eight fixed-coefficient products; there are 16 such trees altogether. Additional pipeline stages may be needed to maintain a cycle time of 25 ns.

A similar study on the effect of partial product rounding on the accuracy of the computation should be carried out to determine the number of partial product bits to be discarded before summation.

Chip characteristics

Table 1 summarizes the characteristics of the chip.

To further characterize the speed of the device, we generated the schmoo plot of cycle time versus supply voltage at room temperature (Figure 7 on the next page). The tester used to characterize the device has a minimum cycle time of 25 ns. The photograph of the die is shown in Figure 8.

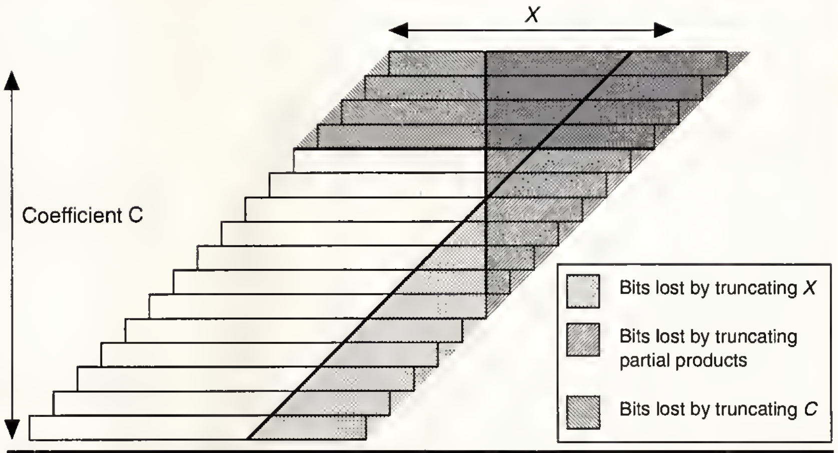

X
Coefficient C
Bits lost by truncating X
Bits lost by truncating partial products
Bits lost by truncating C

Figure 3. Partial product rounding.

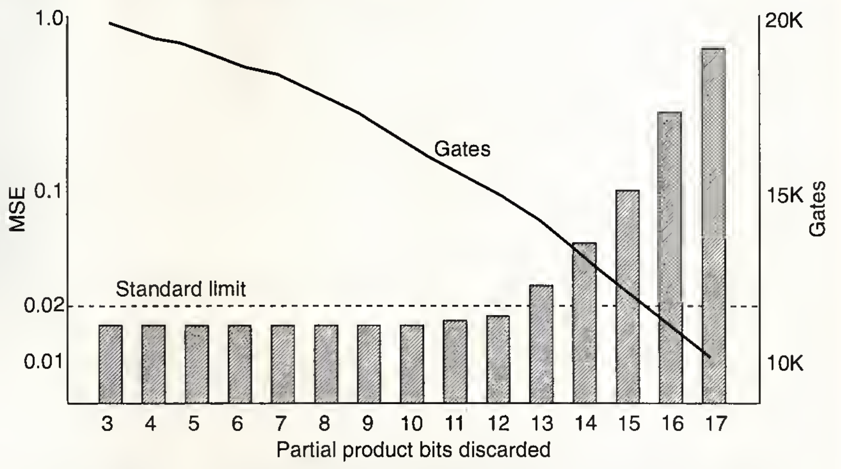


Figure 4. MSE and gate count.

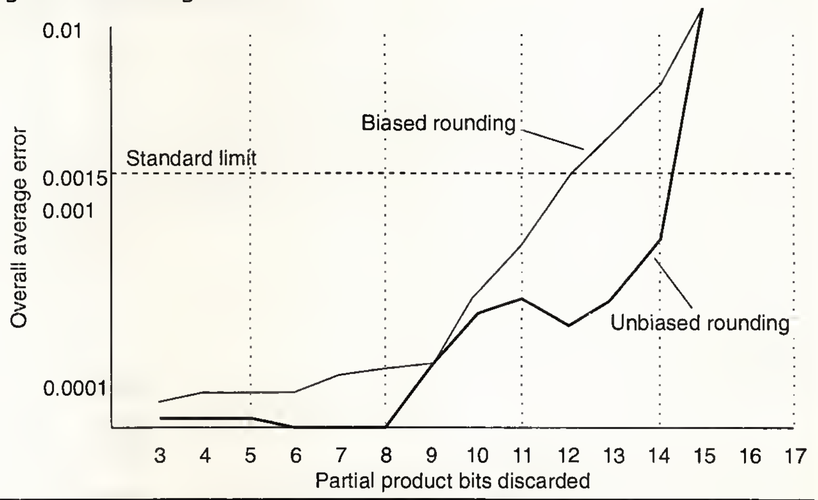


Figure 5. Overall average error.

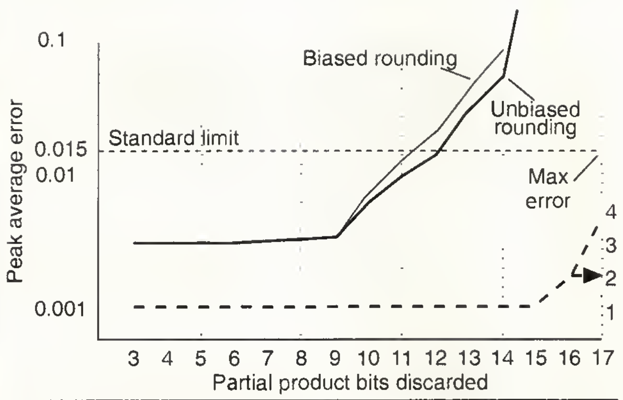


Figure 6. Peak average error.

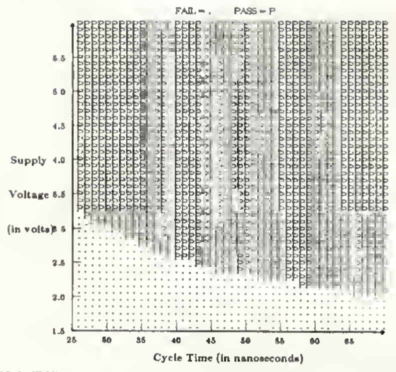


Figure 7. Schmoo plot (room temperature).

Table 1. Chip characteristics of the IDCT chip.

Requirement	Description
Technology	1-micron CMOS gate array
Die size	10.5 mm×10.5 mm
Clock frequency	40 MHz*
Throughput	160 Mpixels per second*
Latency	24 cycles
Package	132-pin ceramic PGA
Gate count	31,000 gates

*Measured at 4.75 volts and 70 degrees C

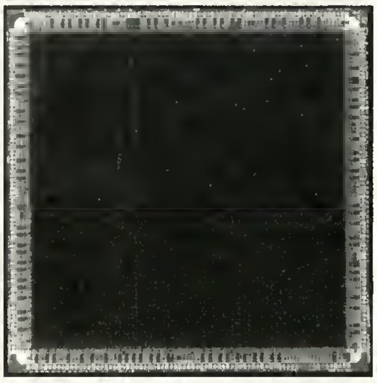


Figure 8. Die photograph.

WALLACE TREE-BASED, FIXED-COEFFICIENT multiplication implemented the 8×8 inverse discrete cosine transform at 160 Mpixels per second. By rounding the partial products before summation and using a simple, unbiased rounding correction circuit after the partial products are summed, we saved over 25 percent of the computational core without impacting the compliance to the scaled CCITT H.261 error specifications for the IDCT in video telephony. We tested the 10.5-mm gate array implementation and found it to be fully functional at a 40-MHz clock rate over a 3.4- to 6-volt range at room temperature.

References

1. S.M. Lei and M.T. Sun, "An Entropy Coding System for Digital HDTV Applications," *IEEE Trans. Circuits and Systems for Video Technology,* Vol. 1, No. 1, 1991, pp. 147-155.
2. M. Barbero, S. Cucchi, and M. Stroppiana, "A Bit-Rate Reduction System for HDTV Transmission," *IEEE Trans. Circuits and Systems for Video Technology,* Vol. 1, No. 1, 1991, pp. 4-13.
3. M. Gottlieb, M.T. Sun, and T.C. Chen, "A Video-Rate 16×16 Discrete Cosine Transform Chip," *Proc. Custom Integrated Circuits Conf.,* 1988, pp. 8.2.1-8.2.4.
4. M. Vitterli and A. Ligtenberg, "A Discrete Fourier-Cosine Transform Chip," *IEEE J. Selected Areas in Comm.,* Vol. SAC-4, No. 1, 1986, pp. 49-61.
5. K.K. Chau et al., "VLSI Implementation of a 2-D DCT in a Compiler," *Proc. 1991 ICASSP,* 1991, pp. 1233-1236.
6. W.-H. Chen, C.H. Smith, and S.C. Fralick, "A Fast Computational Algorithm for the Discrete Cosine Transform," *IEEE Trans. Comm.,* Vol. COM-25, No. 9, 1977, pp. 1004-1009.
7. *Video Codec for Audiovisual Services at px64 Kbits per Second,* CCITT Recommendation H.261, 1990.

Reader Interest Survey

Indicate your interest in this article by circling the appropriate number of the Reader Service Card.

Low 159	Medium 160	High 161

Programmable Vision Processor/Controller

for Flexible Implementation of Current and Future Image Compression Standards

The image compression algorithm standardization process has been in motion for over five years. Due to the broad range of interests that gave input at the national and international levels, the three products of this effort, p×64, JPEG, and MPEG, combine flexibility and quality. The standardization process also included a number of semiconductor companies interested in creating supporting products, which are now nearing completion. One of the first highly integrated products dedicated to video compression available from an IC manufacturer is IIT's Vision Processor/Controller.

Doug Bailey

Matthew Cressa

Jan Fandrianto

Doug Neubauer

Hedley K.J. Rainnie

Chi-Shin Wang

Integrated Information Technology

A number of highly compelling forces drive today's image compression market. Users needing more personal communication continue to force growth of the videophone market, and with the establishment of true industry standards, interest in multimedia computing environments continues to rise. A clearly established market exists for still-frame image transmission to support documentation, news gathering, and faxing. In addition, demand persists for a greater number of high-quality television channels as well as features not currently available from analog TV. In short, a broad range of marketing forces now demands mechanisms for image communication and transmission in the form of complete communication and media packages.

To date, the market has matured sufficiently to produce three industry standards related to various needs in the image compression market:

- *JPEG* (Joint Photographic Experts Group) for still-frame image compression,
- *MPEG* (Moving Picture Experts Group) for compression and decompression of motion video and audio for use in computer systems, and
- *P×64* (also known as CCITT Recommendation H.261) for motion video compression in videophones and teleconferencing applications designed around 64-kilobit/second transmission channels.

Enabling technology

The development of certain key technologies allowed this new market to come about: high-speed silicon with greater levels of integration and mathematical and algorithmic developments in the compression field, such as the discrete cosine transform (DCT). Along with these developments and the resulting feasibility of image compression has come the development of true industry standards—in addition to a number of proprietary ones. These standards make the proliferation of image compression technology possible and commercially practical.

Flexible hardware

Given the number of compression algorithm schemes—both proprietary and industry standard—our company maintains that the only long-term, viable solution is a programmable one. Moreover, many applications now have diverse requirements and call for more than one type of compression. For example, video conferencing may call for the transmission of high-resolution

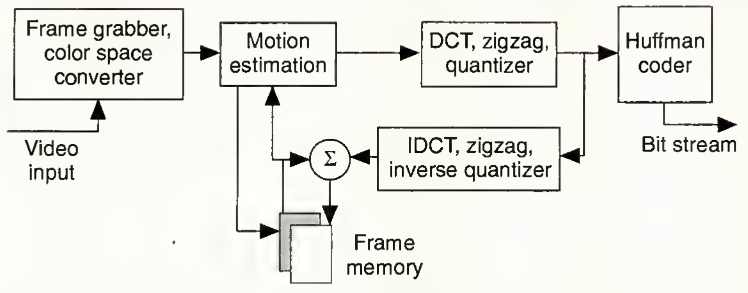


Figure 1. Motion video compression block diagram.

still images over phone lines, suggesting the use of JPEG rather than H.261. Thus a video conferencing system application could require implementation of both standards. Desktop multimedia system applications may require all three standards to be implemented—H.261, say, to communicate video mail to others in the office, MPEG for viewing compact disc movies, and JPEG for still-frame fax, and so on. Such diverse needs make an even stronger case for a programmable solution. Chips or chip sets aimed at particular standards—an approach taken by some image compression silicon vendors—do not allow this flexibility.

Image compression technology

DCT image compression strategies are based on the relative frequency sensitivity of the human eye. Thus, since video data is expressed in pixels, which describe images in terms of the spacial location of their components, pixel data must be transformed to the spacial frequency domain. Figure 1 shows the basic structure of a transform-based, motion video compression scheme.

The first task in most compression algorithms is the conversion of RGB (red, green, blue) to YUV, or luma (Y) and chroma (U and V) values. Since the eye is more sensitive to intensity than to color, the conversion is useful because it separates intensity from color, allowing greater compression to be applied to color information. Thus a fundamental image compression function is the transform matrix that converts RGB values to YUV values.

Once this conversion is complete, the next task is spatial filtering and image resizing. The heart of the algorithm, however, is the DCT that converts the data from one domain to another (space to frequency), that is, YUV values into frequencies. A matrix multiplication implementation of the 2D DCT requires 1,024 multiplications for each 8x8-pixel block. For medium and high image resolutions, designers must build a multiple data path machine.

Following the DCT calculation phase, the algorithm resamples the 8x8 block into a 64×1 linear array. The sample mapping takes place in zigzag order, which clusters the low spatial frequency components at one end and the high frequency components at the other. This step prepares the data for quantization and Huffman coding, which use the frequency clustering to do their jobs more efficiently.

Quantization takes advantage of two facts: that the human eye is more sensitive to low frequency than to high and more sensitive to intensity than to color. Each frequency coefficient is divided by a quantization value, which reduces the number of bits required to express the coefficient. The quantization value can vary over frequency (lower frequency needs a lower quantization value) derived empirically according to subjective quality experiments. This type of image compression is called lossy in that finer differentiation is not regained upon decompression.

In Huffman coding, a more efficient representation of the data assigns the shortest code words to data elements having the highest probability of occurrence. This type of "lossless" compression ensures that no detail is lost upon decompression; only the representation of data changes in the compression process.

Note that, in addition to its DCT, quantization, and Huffman coder blocks, Figure 1 also includes motion estimation blocks, along with last and future reference memory. This particular diagram describes the MPEG compression process but is in fact a superset of the circuit blocks required for the JPEG and P×64 standards. While JPEG is aimed at still-image compression and decompression and requires no motion estimation algorithm, P×64 uses predictive interframe encoding for motion estimation. MPEG uses both predictive interframe encoding and interpolative interframe coding, as described later. MPEG represents the most complex scheme. Note that MPEG requires both past and future frame reference memory for motion estimation, while P×64 motion estimation needs only past memory.

The motion estimation aspect of video compression attempts to reduce the amount of data required to represent an image by comparing frame-to-frame variations in the video sequence. When the motion estimation block finds that the pixels have changed, it searches the location nearby for an area that best matches the new pixels. If it finds a good match, it sends the vector to the matching pixels instead of the pixels themselves. This step can save a large number of bits in the transmission channel.

For both P×64 and MPEG, the motion estimation process starts with an intraframe, a single frame coded independently of any other frame in the sequence. During video compression new intraframes or intracoded macroblocks usually transmit intermittently to ensure the integrity of the process, since lossy interframe compression can eventually degrade the image due to different mathematical performance of the encoder and decoder.

When one frame is coded in terms of another frame, the encoding process is called interframe encoding, two types of which are predictive interframe and interpolative interframe coding.

Predictive interframe coding begins with an independently coded intraframe; then the algorithm compares each subsequent frame to its predecessor and codes each in terms of differences.

Interpolative interframe coding processes two nonadjacent basis frames (past and future)—using either intraframe coding, predictive interframe coding, or a combination— then constructs intervening frame images by estimating the motion and stepwise changes between the two basis frames.

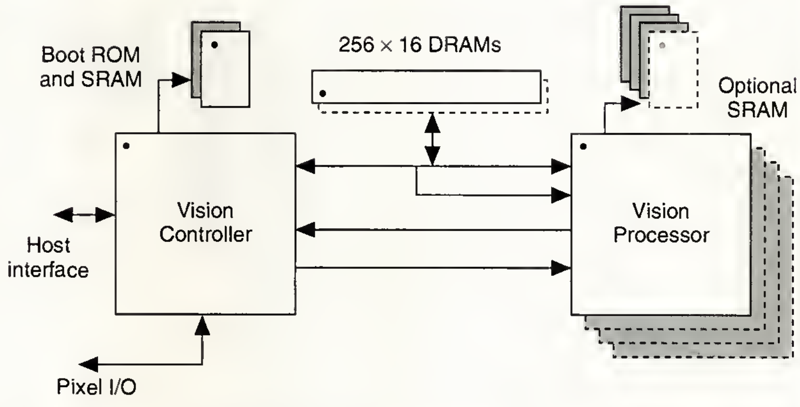


Figure 2. Image compression subsystem.

Programmable chips

Two chips perform the variety of functions just described: the Vision Processor (VP) and the Vision Controller (VC). A basic image compression subsystem requires one of each of these and a small number of memory chips. Designers can connect multiple VPs to one VC for higher performance. Figure 2 shows a typical connection of the VP and VC chips for multistandard compression and decompression.

In Table 1 the one VP/one VC system can act as an H.261 QCIF (Quarter Common Interchange Format, 176×144 pixels at 30 Hz) video codec for videophone applications and can decode full CIF (352×288 pixels at 30Hz) in real time. For full-CIF, 30-frame/second, encode performance designers must add an additional VP to allow for the additional motion estimation load. The one VC/VP system can also perform MPEG SIF (Standard Interchange Format, 352×240 pixels at 30 Hz) decoding at 30 frames/s and real-time MPEG encoding, taking advantage of the algorithm's flexibility to reduce the processing load at the expense of high quality. High-quality MPEG encoding, using optimal coding modes, can be processed at a reduced frame rate of 10 frames/s.

The Vision Processor. The VP (Figure 3 on the next page) forms the heart of the image compression system. It can perform DCT, quantization, and motion estimation, as well as inverse DCT, and inverse quantization. The highly parallel and microcode-based processor performs all of the JPEG, MPEG, and Px64 algorithms. It includes a custom RISC engine, and IIT supplies the microcode for the algorithms.

The VP has a 64-bit, parallel architecture, including fast multiply/accumulate circuitry, sixteen 8-bit ALUs, as well as a shifter and tree adder for motion estimation. Transfers of pixel data and run/amplitude tokens to and from the VP pass through the VP's high-speed DMA data port; it also has a 4-Kword ROM for storage of microcode instructions. Currently, 25- and 33-MHz versions are available.

Table 1. Compression and decompression capabilities of a 1VC/1VP circuit.

Algorithm	Resolution	Pixels	Encode/ decode	Frame rate (fps)
H.261	QCIF	176×144	Codec	30
H.261	CIF	352×288	Decode	30
H.261	CIF	352×288	Encode	15
MPEG	SIF	352×240	Decode	30
MPEG	SIF	352×240	Encode	Up to 30
JPEG	Variable	Variable	Decode	Variable
JPEG	Variable	Variable	Encode	Variable

The Vision Controller. The VC (Figure 4) smart microcontroller controls the compression process and provides the interface to the host system. It captures pixels from a video source, performs video preprocessing, supervises pixel compression by the VP, performs Huffman encoding, and passes the compressed data to the host over a buffered interface. Simultaneously, it can take compressed data from the host, perform coder decoding, supervise decompression via the VP, perform postprocessing, and generate digital pixel output for a video destination such as a monitor.

The VC contains a 32-bit RISC (reduced instruction-set computing) microcontroller with on-chip cache, which is conveniently programmable in C. Note that the VC contains those functions that the system designer would most likely want to customize or modify for a particular application.

The VC converts color and space values in both directions: RGB to/from YUV in a special, on-chip circuit block. The VC also has a programmable CRT controller, which can directly drive a triple digital-to-analog converter for monitor display. In addition, text and graphics can be overlaid onto the video,

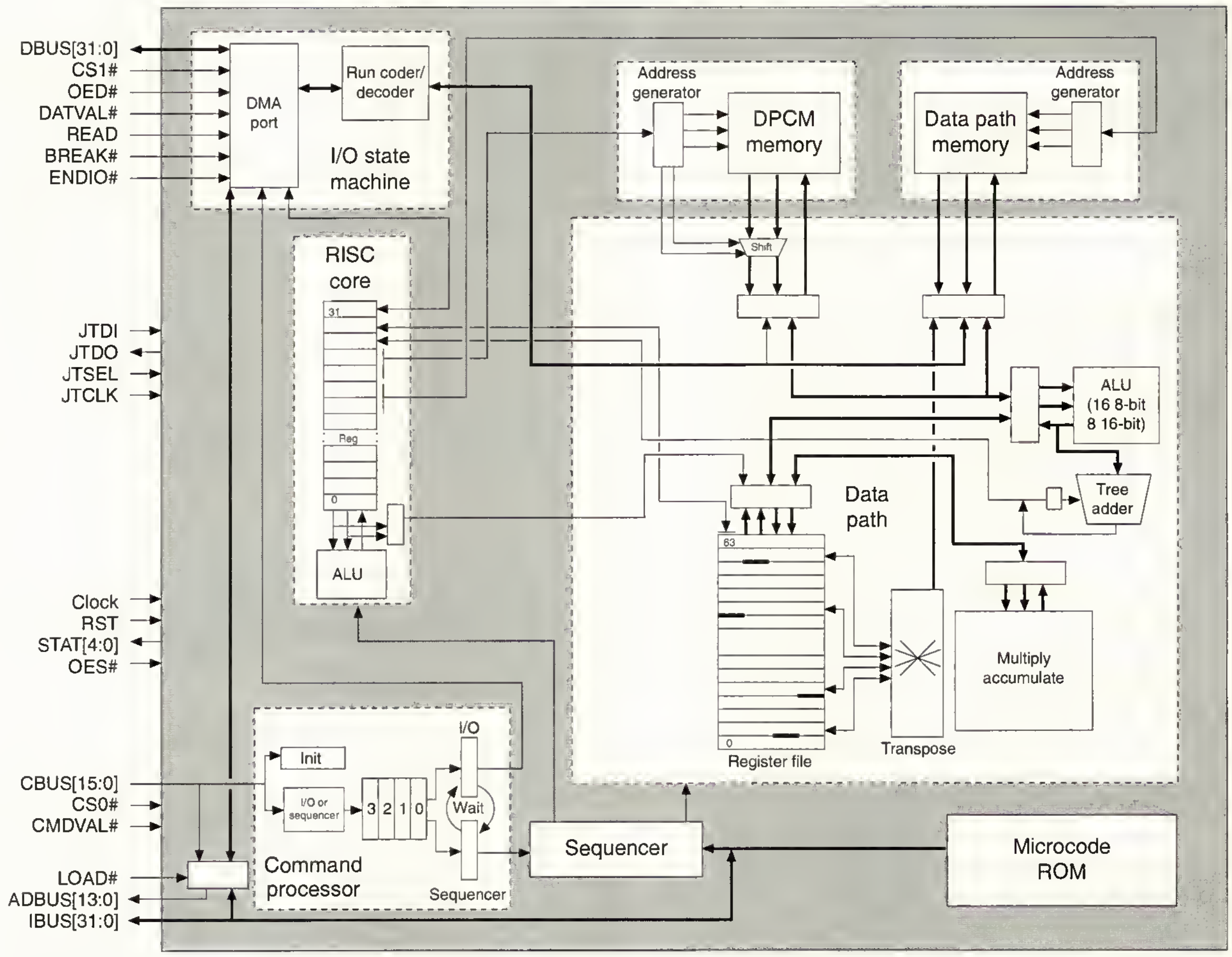


Figure 3. Vision Processor.

under control of the RISC microcontroller. The video timing section can also genlock (synchronize the video timing) to other video sources such as video cameras, VCRs, or VGA displays to help with Graphics Overlay and Picture in Picture (PIP) features.

A special circuitry in the VC handles required up and down scaling processes. Scaling is necessary because most digitized video sources supply images that are larger than those used by the MPEG and H.261 algorithms. For instance, another standard (CCIR Rec. 601-2) specifies an image resolution of 720×240 pixels at 60 fields per second, which must be scaled by a factor of four to be suitable for MPEG at SIF resolutions. The scale-down circuitry can reduce the resolution of the incoming video data in two distinct ways. First, the U and V components can be scaled by a factor of 2:1 in both horizontal and vertical dimensions. In addition, the Y, U, and V components may all be scaled by an additional factor of 2 in the horizontal direction only. The scale-up circuitry has two functions for increasing the output frame size. The U and V components can be zoomed up by a factor of 2:1 in both directions, and an additional factor of 2:1 may be applied to the Y, U, and V prior to output on the pixel bus.

The VC contains a temporal filter that is especially useful in videophones. When noise occurs due to image subsampling, motion, or other event, the compression algorithm interprets the noise as motion in the image. The temporal filter helps to reduce the noise that reaches the coder and thus improves the quality, especially at low bit rates.

The VC is a full codec, since it contains sufficient circuitry to both code and decode with the same chip at the same time at QCIF resolution. Figure 4 shows the Vision Controller block diagram.

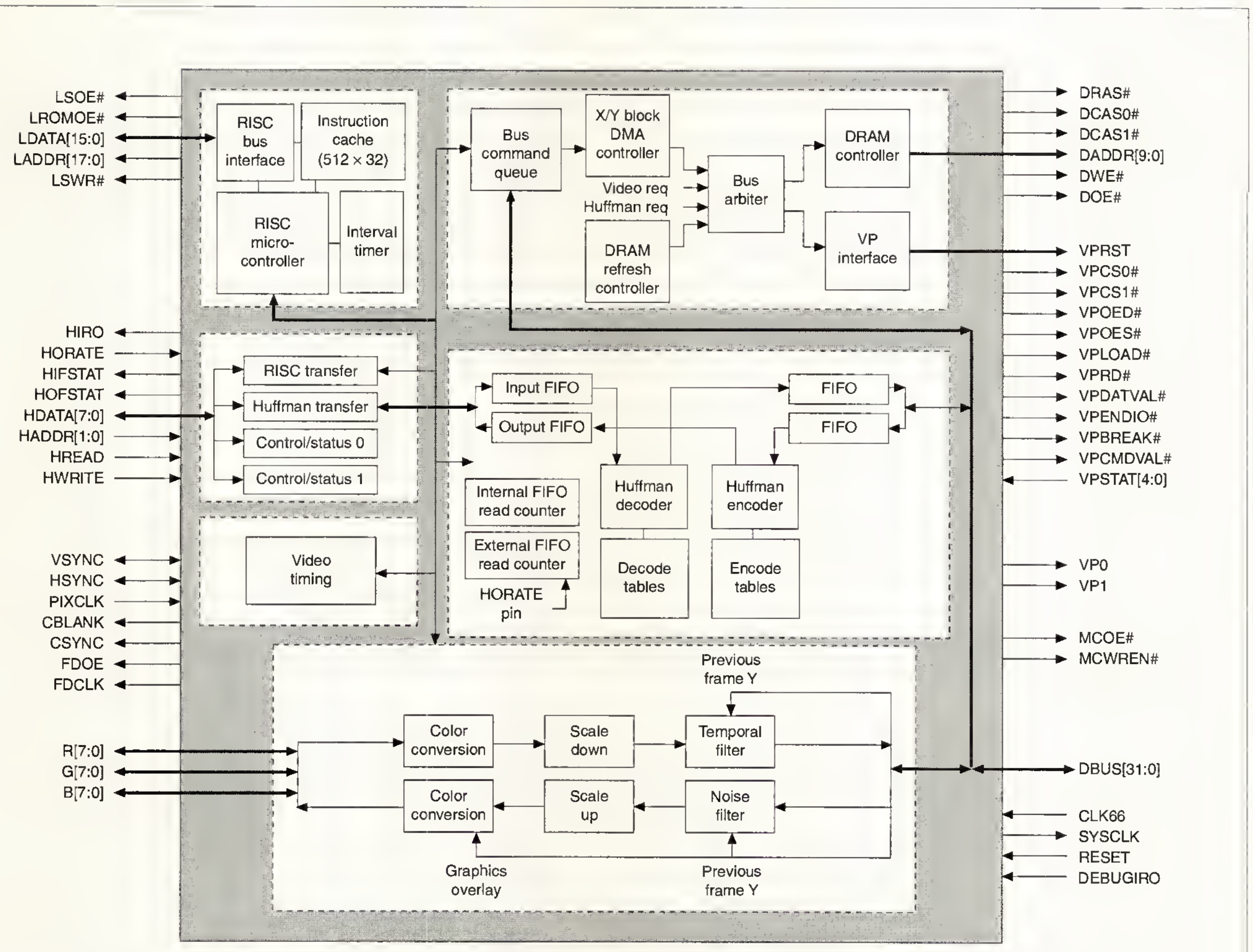


Figure 4. Vision Controller: detailed block diagram.

DCT and motion estimation implementation

The VP contains the means of accomplishing DCT, inverse DCT, quantization, inverse quantization, and motion estimation. For an image compression system, these are the computation-intensive calculations that must be done in real time. Thus the VP contains structures that allow these calculations to be performed at high speed.

Mapping a full-search motion estimation algorithm onto the VP. For motion estimation in P×64 applications, data transfers from the VC to DRAM and then onto the data port of the VP—all under control of the VC. Note that the data is buffered in frame memory DRAM until it is required.

The VC stores two frames of memory in the DRAM, new and old. It takes the current 16×16 block from new frame memory and stores it in the DP memory in the VP. Then it copies the 32×32 search window from the old frame into DPCM memory.

Somewhere within it, the 32×32 old frame block may contain a shifted version of the 16×16 new frame block. This example uses a search range, in both planes, of +7, –8; and thus the 16×16 new frame block is positioned in the 32×32 old frame block, so it can come up with the best possible match.

The following formula calculates the sum of differences, or dispersion value, for a particular (x,y) location of the new frame within the old frame:

$$D(x,y) = \sum_{i=0}^{15} \sum_{j=0}^{15} ABS\left[\text{Old}(i+y, j+x) - \text{New}(i,j)\right]$$

For each value of x and y, $D(x,y)$ represents a sum of differences between the new frame and old frame. Thus a 16×16 block of dispersions is generated, along with associated vectors used for motion estimation; the smallest dispersion value of $D(x,y)$ indicates the best matched location of

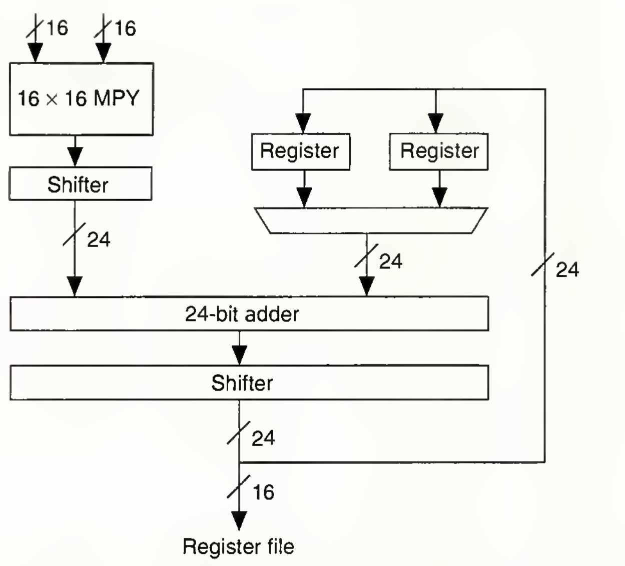


Figure 5. Part of multiply/accumulate data path of the Vision Processor.

the block. A number of statistical calculations can be performed on the data to determine if $D_{\min}$ represents a valuable match, or whether the block has moved out of the search range or changed too much for the match to be useful.

To actually carry out this computation, the microcode simultaneously issues a motion compensation (MC) command to the sequencer while loading the required memory addresses into specialized RISC registers. Data from the DP and DPCM memories moves to the ALUs for subtraction and then to the tree adder, which accumulates the data; the answers are replaced in DPCM memory.

One line at a time is subtracted or "differenced" by the ALUs, then accumulated by the tree adder and stored there until an entire block is complete.

Due to its flexibility, the VP can perform less computation-intensive or more elegant algorithms. A two-step algorithm lets every other pixel be used for differencing; hence, x and y are incremented by 2 rather than 1. Then a local full-search is performed in the area of the minimum dispersion.

Once the motion vector has been found, the difference block can be calculated from the original new block and the correct window in the old block. The difference block can then move to the data path for the DCT. It is worth noting that while the data path processes the MC (block differencing) command, it can be performing the MPY and MAC commands required by the DCT of a previous block. Motion estimation is only performed on the intensity component (Y).

Mapping a full-matrix multiply 2D DCT onto the VP. This process downloads source data from the DP or DPCM memories into a 64-bit-wide data path register file. The data is arranged to give each pixel 16 bits of the 64-bit width of the register bank. Four MACs (multiply accumulators) can operate simultaneously in the data path (illustrated in Figure 5). MAC results return to the register file when accumulation is complete. The intermediate storage is 16 bits, and programmable shifters manipulate the relative significance of the product, accumulator value, and output results. The dynamic range of the mathematical elements is more than enough to perform the DCT to the CCITT H.261 specification. The general structure of the 1D DCT equation is

$$F(m) = K(m)\sum_{n=0}^{7} f(n)\,C(m,n)$$

$C(m,n)$ and $K(m)$ are stored constants, $f(n)$ is the eight-element set of pixels, and $F(m)$ is the set of output coefficients. Both m and n go from 0 to 7.

With up to four blocks downloaded into the register file (from DP to DPCM memory), the required MAC operations for the DCT can be performed. The user initiates the DCT command by passing the address of the starting point of the DCT routine stored in microcode to the sequencer. The microcode contains a field with a 16-bit value used for the cosine transform multiplication values as well as a field for the instruction that is being executed (multiply immediate). The data moves into the multipliers at the same time, as does the pixel data. Four columns of each block change simultaneously, processing 32 pixels in eight cycles or one block in 16 cycles. Then the result is transposed, and the process repeats for the complete 2D DCT. The final output of the MAC operation is stored back in the data path registers.

Each one of the multipliers has an associated accumulator, allowing the results to be added and saved. After multiplying/adding eight lines, one part of the DCT block has effectively been done. The answer returns to the register file; then, because a 2D DCT is required, the transposer transposes blocks (rows to columns, columns to rows).

VC software interface

The VC contains a RISC microprocessor core programmable in C, and IIT supplies the compiler and debugger to aid in speedy algorithm implementation or modification. A side benefit of this approach is that the user can view the VP as a high-speed accelerator for the VC C code, simply calling the high-speed routines as required. In addition, a software interface for the VC helps all the standard algorithms (MPEG, JPEG, and P×64) to allow users to access the power of the VP/VC system with little or no software development. For example, the MPEG decoder software interface allows the user to send commands, such as fast forward, fast reverse, slow motion, and so on, making the system very easy to integrate into multimedia disk player operating systems.

The encoder interfaces allow full access to the many features of the various standards. This capability gives the user a high degree of flexibility in encoding strategy, one not available in hardwired chip solutions. The user can trade off the many factors affecting encode quality, allowing tuning of the algorithm for specific applications. In summary, the user can program the VC, or modify existing code, taking advantage of any specific art in encoding and building in system flexibility for the future.

CURRENT VERSIONS OF THE VP AND VC are available at 25 and 33 MHz; 40- and 50-MHz parts are planned for the future. Software improvements will include improved quality for the standard algorithms. As device speed increases, computing power becomes available for additional calculations such as image statistics gathering to improve quality, and even audio compression and decompression.

Bibliography

F.A. Kamangar and K.R. Rao, "Fast Algorithms for the 2D DCT," *IEEE Trans. Comput.*, Vol. C31, 1982, pp. 889-906.

A. Rosenfield and A.C. Kak, *Digital Picture Processing*, Vol. 1, Academic Press, 1982.

C.P. Sandbank et al., *Digital Television*, J. Wiley and Sons, Chichester, England, 1990.

Doug Bailey is senior applications engineer for Integrated Information Technology where he holds support responsibility for the video compression product line. He previously worked for LSI Logic Corporation on its P×64 videophone chip set and the Inmos DSP Group on the design of the IMS A121 DCT processor. Bailey holds a BEng degree from the University of Birmingham, England. He is a member of the Institute of Electrical Engineers in the UK.

Matthew Cressa is a member of the technical staff at IIT. Prior to this, he worked at Poqet Computer Corp. on low-power ASICs for hand-held computers. He holds a BSEE from Santa Clara University.

Jan Fandrianto, director of engineering at IIT, works on floating-point and video processors and digital signal processing. Prior to this he served as engineering manager at Weitek Corp. Fandrianto holds a BSEE from the University of California, Berkeley, and an MSEE from Stanford University. He is a member of the IEEE.

Doug Neubauer, video hardware design manager at IIT, heads the board design activity and supports video implementation efforts. He has worked for National Semiconductor, as well as Hewlett Packard, Atari, and QMS. Neubauer holds a BSEE from Oregon State University and is a member of the IEEE.

Hedley K.J. Rainnie is software engineering manager at IIT working on vision products. He previously worked at Imagen on laser printers and at Philon on compilers, and was a consultant to Allied Signal and NYU Brain Research Lab. He holds an MS in computer science from New York University and is a member of the Association of Computing Machinery.

Chi-Shin Wang founded Integrated Information Technology in 1987 and serves as its president. He also founded Weitek Corp. in 1981 and was vice president of engineering and R&D. Wang holds an MS in physics from the California Institute of Technology in Pasadena and a PhD in electrical engineering from Stanford University. He is a member of the IEEE.

Direct questions about this article to Doug Bailey, Integrated Information Technology, 2445 Mission College Blvd., Santa Clara, CA 95054; or e-mail to bailey@iit.com.

Reader Interest Survey

Indicate your interest in this article by circling the appropriate number on the Reader Service Card.

Low 162	Medium 163	High 164

The V.42bis Standard for Data-Compressing Modems

The recently adopted CCITT V.42bis standard for data-compressing modems is a conservative and economically implementable scheme, one discussed here from algorithmic, experimental, practical, and marketing standpoints. V.42bis compresses text about as well as the Lempel-Ziv-Welch algorithm of the Berkeley Unix Compress utility; other Ziv-Lempel variants are discussed briefly. Even though better compression ratios can be obtained with these variants, V.42bis is eminently suitable for implementation on a contemporary modem.

Clark Thomborson

University of Minnesota at Duluth

Many factors can affect modem throughput. Important factors include the protocols used for data compression, telephone signaling, error correction, and computer-to-modem transmission. To overcome some of these problems, the CCITT V.42bis standard details a textual substitution scheme that allows a well-implemented modem to achieve a 2.3:1 compression ratio on English text. A modem implementing this standard delivers higher bandwidth, on compressible files, than a noncompressing modem.

Two major V.42bis modem applications are terminal-to-computer and computer-to-computer communications over the telephone network. However, text transmission is only a fraction of current computer communication and this fraction is rapidly decreasing. Better algorithms for compressing text (and other types of data) are being developed, but I believe V.42bis will become the dominant method for text compression for at least the next 10 years.

Background

Briefly, the V.42bis data compression method is a textual substitution scheme based on Ziv and Lempel's second algorithm.[1] V.42bis incorporates Welch's coding scheme[2] as well as Miller and Wegman's[3] (and, independently, Storer's[4,5]) idea for incremental modification of a full dictionary using a least recently used (LRU) heuristic. V.42bis also has an efficient cleartext escape mechanism, differing in implementation but not in spirit from that proposed by Fiala and Greene.[6] (See the Standards box for more information.)

In 1990 users could buy a modem with the following characteristics:

- Price: $670 (discounted) to $1,150 (list)
- Modem-to-terminal or -computer connection: EIA-232-D (300 to 38,400 baud)
- Modem-to-telephone network connection: V.32 (300 to 9,600 bps)
- Error correction: V.42 (16-bit CRC with retransmission)
- Data compression: V.42bis (modified Ziv-Lempel)
- Implementation: 8-bit microprocessor (10-MHz Z80) with a 40-Kbyte RAM and 64-Kbyte ROM.

I will call such a device a V.42bis modem, although I caution the reader that other uses of the V.42bis standard are possible. The implementation and pricing information correspond to a MultiTech MT932EAB, introduced in late July 1990. Other manufacturers offered six similar products, according to an informative set of product reviews.[7]

Standards

I encourage expert readers to look at Byrd's article on V.42bis modems[7] and/or Black's discussion of the EIA-, X-, and early V-series standards.[8]

Unfortunately, standards documents are very difficult to read, even for an expert in the field. Even more obtuse, but nonetheless important, are the US patents granted in the last decade for data compression methods. These patents appear to cover implementations of many of the algorithms I discuss. Michael Ernst of MIT, mernst@theory.lcs.mit.edu, has collected a list of such patents. If you have Internet access, you may retrieve his list by anonymous ftp from mintaka.lcs.mit.edu: /mitlpf/ai/patent-list.

One year later, the first V.32bis modems became commercially available. All such modems include the V.42bis data compression feature, so it is superfluous to call them V.32bis/V.42bis modems. The advantage of a V.32bis modem over the older V.42bis modem is that the V.32bis protocol provides higher (14,400 bps) raw bandwidth on the telephone network than the older, slower 9,600-bps V.32 protocol of a V.42bis modem. A V.32bis modem can deliver 50 percent greater throughput than a V.42bis modem. However, as noted in the adjacent Terms box and elsewhere, telephone network bandwidth is only one of many possible bottlenecks.

PC World's December 1991 product reviews[9] listed V.32bis modems at $425 to $1,345, while V.42bis modems cost from $399 to $1,395. The wide range of pricing is remarkable. The name-brand manufacturers charge the higher prices for their latest models. Available, however, are deep discounts, especially on discontinued lines. One distributor (Damark) of-

Terms

baud: The bandwidth of a communication channel, expressed in units of symbols per second. Baud is frequently assumed to be equivalent to bps, but this is not always true. If the symbols are taken from a binary alphabet, 1 baud is indeed 1 bps. However, if the symbols are taken from a 16-valued set, each signal carries four bits, and 1 baud is 4 bps.

BCH coding: Any of a variety of error-correcting codes based on work by Bose, Ray-Chaudhuri and Hocqeunghem. Reed-Solomon codes form a related class. These codes require much more computational power than the simple CRC-check-and-retransmit method of V.42. However, they impose much less overhead on the communication channel. Some of these codes can correct huge numbers of errors without retransmission, making them invaluable in satellite communication and other applications where errors are frequent and retransmission is infeasible.

bis: A suffix indicating a modified standard, as in V.32bis or V.42bis. Derived from the Latin *bis* meaning twice.

bps: Abbreviation for bits per second, a commonly used unit for expressing the bandwidth of a communication channel.

CCITT: Consultative Committee on Telephony and Telegraphy, a Geneva-based division of the International Telecommunications Union, a New York-based United Nations organization.

cleartext: Uncompressed data, often assumed to be English words in ASCII code.

cps: Bandwidth of a communication channel in (8-bit) characters per second. In this article, cps figures include the overhead of timing and error control. Moreover, they are adjusted for data compression, so an 8-bps connection provides somewhat more or less than 1 cps of effective bandwidth.

CRC: Cyclic redundancy code, usually a 16-bit signature obtained by a simple hashing algorithm on keys with hundreds of characters. CRC forms the basis of V.42 error control: A modem sends a few hundred characters (or less, if there is not much data to send), followed by a 16-bit CRC. The receiving modem computes a CRC on the data characters it receives, comparing it to the CRC it receives. If the two CRC values are the same, it is unlikely that any transmission errors occurred.

EIA-232-D: An internationally recognized standard for the electrical and mechanical connections between a terminal and a modem, or between a computer and a modem. Data transmits bit-serially, typically in an asynchronous format of 8-bit characters with 2 or 3 timing bits per character. Other formats are possible. The maximum recommended bandwidth is 19,200 baud, or about 1,920 cps in an asynchronous format. Most V.42bis modems, and some high-speed terminals and computers, will communicate reliably over an EIA-232-D connection at 38,400 baud, or 3,840 cps. Some modems can communicate at 57,600 baud using an EIA-232-D connection to a special board that plugs directly into an IBM PC bus. This higher priced option, using a synchronous format for character transmission with little timing overhead, can provide 6,000 cps of bandwidth.

hash trie: A trie in which each node contains just one pointer, referencing the node's ancestor. The nodes are stored fered a V.42bis modem, with limited supply, for $270. The overlapping price ranges for V.32bis and V.42bis modems indicate that the V.32bis modem is well on the way to supplanting the V.42bis modem, just as the V.42bis modem recently supplanted an earlier, non-V.42bis generation of high-speed modems.

Modem applications, performance, pricing

Terminal-to-computer and computer-to-computer communications employ the telephone network, either with a dial-up connection or a leased line.

The distinguishing feature of a V.42bis modem is that it can deliver higher bandwidth, in most applications, by the use of a good algorithm for data compression. Experts may wish to read the V.42bis Compression box on the next page for detailed information on this algorithm. Nonexperts need know only that the performance of a V.42bis modem will vary with the compressibility (or incompressibility) of the data being transmitted. For example, approximately a two-fold improvement is typical in an application involving English text. Thus, when transmitting English text, a V.42bis modem will deliver somewhat more than 2,000 characters per second, up from about 1,000 cps on a V.42 modem. If relatively incompressible data is being presented to the V.42bis modem, or if some bottleneck exists in the system other than the modem, the V.42bis modem will not deliver performance improvement. On highly compressible data, such as spreadsheets, V.42bis data compression can deliver up to a fourfold improvement in bandwidth. Will this bandwidth improvement be achieved in a given application? Read on to find the factors, other than data compressibility, that can limit modem performance.

User's desired bandwidth. In a terminal-to-computer application, a user typically accesses a remote computer with a terminal in an office or home. All the user's keystrokes are

continued on p. 46

Terms (continued)

in an array, allowing rapid access if a node's index is known. Searching for a string in a hash trie involves hashing the string to find a set of possible indices for its node, then testing (by traversing the ancestor chains) whether one of the indexed nodes represents the input string.

hashing: The digital scrambling of a key (a series of data bits) into a signature (a shorter series of bits). The best hash functions give distinct signatures for almost all commonly encountered keys.

modem: A device performing the modulation/demodulation function necessary to transmit digital signals over an analog connection such as a telephone line.

Patricia trie: Binary trie in which each node contains an integer index and two subtree pointers. The integer is the index of the leftmost bit in which all keys in the left subtree differ from all keys in the right subtree. Complete keys are stored at the leaves of the tree.

pointer-list trie: A trie in which each node contains pointers to siblings (nodes at the same level), one or more descendants, and possibly to that node's ancestor. Searching for a string in a pointer-list tree involves the traversal of sibling lists and the following of descendant pointers.

trie: A tree-shaped data structure for the storage and reTRIEval of elements of a set of character strings. Typically, each node in a trie contains one character of data, namely the first character of all the strings representing that node's descendants. A trie node must also contain one or more pointers to other trie nodes, to define the connectivity of the tree and let the trie be traversed efficiently.

V-series interface: Any of a number of protocols for data communications over the telephone network.

V.32: An internationally recognized standard for sending digital information over a telephone connection with a modem. The maximum bandwidth is 9,600 bps. Each signal carries 4 bits; the signaling rate is 9,600/4 = 2,400 baud.

V.32bis: A recent revision of V.32, allowing 14,400 bps.

V.42: A standardized method for error control over a V-series interface, providing for retransmission of information in case of error. This protocol typically has about 12 percent overhead, according to my MultiTech manual, so a 9,600-bps V.32 connection with V.42 error control has an effective bandwidth of (9,600/8)/112 percent = 1,060 cps. The overhead of the V.42 protocol will be larger on a noisy telephone line, for example, on an overseas connection, so the available bandwidth will actually be less than 1,060 cps.

V.42bis: A standardized method for data compression on a V-series interface. Single-bit errors can have catastrophic results, so V.42bis is normally used in conjunction with the V.42 method for error control. On English text that is compressible at 2.3 to 1, a 9,600-bps V.32 connection with V.42bis has a bandwidth limit of (2.3) * (9,600/8)/112 percent = 2,440 cps. A 14,400-bps V.32bis connection with V.42bis data compression has a bandwidth limit of about (2.3) * (14,400/8)/112 percent = 3,700 cps on English text. Other types of data give rise to different compression ratios, and thus to different bandwidth limits.

V.42bis compression

V.42bis is a textual substitution scheme based on Ziv and Lempel's second algorithm. Whenever a code word is transmitted, the dictionary is augmented by a new code word consisting of the code word being transmitted, concatenated with the next character in the input stream. Initially, the dictionary consists of all one-character strings, in ASCII order.

For example, if the string *ababababa* is transmitted, the first code word is the ASCII code for *a*. The second code word is the ASCII *b*. After this second transmission, both the receiver and the transmitter know what character followed the first code word, so both add the string *ab* to their dictionaries in slot number 256. This newly entered code word is immediately useful, since it can be transmitted (as a 9-bit value) as a representation for *ab*. Now both modems enter *ba* into their dictionaries, in slot number 257: The general rule is to concatenate the first character of the just-received code word to the string represented by the previous code word. The fourth transmission is another 256 (*ab*), whereupon both modems enter *aba* into their dictionaries in slot number 258. The fifth transmission, of code word number 258 (*aba*), completes our example.

It is not hard to prove that strings with repeated substrings are efficiently coded by this procedure. English text is moderately repetitive: Some substrings, such as *th* and *the* occur much more frequently than other substrings, such as *qz*.

As just indicated, V.42bis transmits only previously created code words, expressing these as binary strings with lengths that are the logarithms of the current dictionary size. This is a relatively new idea, due to Welch. (Ziv and Lempel suggested transmitting a just-created code word by sending a previously created code word index followed by an unencoded character. A Welch-style output stream contains only code word indices, which is to say, only compressed data.) The Unix 4.3 bsd Compress utility is a highly tuned implementation of Welch's algorithm. For this reason, it is often referred to as an LZW algorithm, after Lempel-Ziv-Welch.

Welch introduced another refinement, a clever method for adding code words one step earlier than just described. The V.42bis standards committee decided not to adopt this method, thereby simplifying and accelerating the decoding process slightly. Their experiments showed little cost in compression ratio, or in encoding speed.

The V.42bis standards committee was heavily influenced by the ideas in Alan Clark's 1987 doctoral thesis from Leicester Polytechnic. For example, V.42bis makes room for a new code word in a full dictionary by deleting a code word that is not a prefix of any other code word. V.42bis finds candidates for deletion by cycling through the dictionary. This process of deleting old code words is similar to that proposed independently (and earlier) in Storer's least recently used dictionary updating scheme and in the Miller-Wegman code.

V.42bis uses a two-character sequence, beginning with the special character <esc>, to shift into transparent mode whenever this would provide greater compression (or less expansion). A V.42bis transmitter can restart compressed-mode transmission by sending another two-character escape sequence at any subsequent time. This is somewhat like the literal mode of Fiala and Greene's Ziv-Lempel variant. However, Fiala and Greene include the length *l* of the literal string with the escape character sequence. Their algorithm must therefore "look ahead" in the input stream to determine an appropriate *l*, to use transparent mode efficiently.

V.42bis updates the dictionary during transparent mode, just as it does during compressed mode, so that the dictionary is always adapting to the incoming data.

To avoid undue expansion during transparent mode, V.42bis changes the identity of <esc> every time this code is encountered in the input stream, using the formula <esc> = (<esc> + 51) mod 256. This clever scheme will find an unused character in the input stream to serve as an escape, if there is such a character. Thus any file that doesn't contain all 256 character codes can be encoded, in transparent mode, into a file that is only slightly longer than the original.

If the input does contain all 256 character codes, but the input characters are uncorrelated with the escape character update function, any given character transmission has only one chance in 256 (on long-term average) of being encoded as the two-character sequence <esc><eid>. An upper bound on the expected expansion for uncorrelated input in transparent mode is thus $1 + 1/256 = 1.004$. Of course, a pathological input can be constructed in which the character sequence tracks the escape code update function, leading to a worst case expansion of 2.0. This last case is unlikely to occur in any application, so I conclude that essentially no expansion occurs in the transparent mode of V.42bis.

V.42bis also provides an escape-code sequence that causes the current dictionary to be discarded. This sequence, as well as the transparent-mode escape, can be transmitted by the compressor at will.

Algorithmically, V.42bis is underspecified, if only for the reason indicated in the previous paragraph. A V.42bis-com-

V.42bis compression (continued)

pliant modem need not provide data compression at all, when it transmits. It might never leave transparent mode, or it might enter compressed mode only when expansion would result. A more likely scenario is that some V.42bis modems will have somewhat better compression characteristics than others, due to differences in their heuristics for deciding when to send escape codes.

Another area of algorithmic underspecification in V.42bis lies in its openness regarding two key parameters, the maximum dictionary size and the maximum code word length. When a V.42bis connection is first established, the calling modem and the answering modem negotiate these values. A fully compliant, but underconfigured, V.42bis modem can insist on the minimum values of 512 code words with at most six characters apiece.

In both the V.42bis implementation suggested by Alan Clark and in the sample source code used by the V.42bis study group, a dictionary is implemented as a pointer-list trie with one node per code word. A node in such a trie contains two pointers (lchild, rsibling) and one 8-bit ASCII character. A pointer requires 2 bytes on an 8-bit microprocessor with a 16-bit address space. The trie nodes can be stored in a dense linear array, so the string corresponding to a node can be uniquely specified by transmitting the index of that node in the array. Note that 16-bit node indices could be used instead of pointers, if this is more efficient, transforming the structure into a cursor-list trie.

To permit rapid deletion of leaf nodes, these methods allow a third parent pointer to be added to each record. Alternatively, an additional 8-bit unsigned integer would suffice. This integer could indicate the depth of a trie node, since the V.42bis standard limits code words to a length of at most 250 characters. The rsibling field of the rightmost child could then contain a parent pointer. This pointer could be distinguished from a normal rsibling pointer by a change in the depth of the node being referenced.

By this argument, a V.42bis dictionary can be implemented as a pointer-list trie, using just 6 bytes per node. A minimally compliant (512-code word/dictionary) V.42bis modem thus requires just 3 Kbytes of data space to hold its dictionary. Such a modem could only compress in one direction of communication, as would be appropriate on a home terminal: The data can be displayed at high bandwidth, but keyboard input is always transmitted at low bandwidth.

My MultiTech MT932EAB manual does not specify the maximum size of its dictionaries. However, physical inspection shows it to have just 40 Kbytes of RAM. Since this modem supports bidirectional data compression, it can use at most 20 Kbytes for each of its dictionaries. In fact, it cannot use even this much RAM for its dictionaries, since many Kbytes are required for data buffers in any modem implementing V.42 error correction. Using the minimum of 6 bytes for each V.42bis dictionary entry, then, I find that a 2,048-code word dictionary might be feasible in an MT932EAB. A 4,096-code word dictionary, however, would be infeasible for bidirectional transmissions.

The MT932EAB modem did well in a trade magazine's throughput tests,[7] indicating that its compression ratios were competitive despite its limited dictionary size. This is not surprising, since my experimental data shows that V.42bis does a good job of compression even when limited to a 512-code word (3-Kbyte) dictionary.

The expert reader might wonder why a hashed data structure is not appropriate for V.42bis. Indeed, such a structure is feasible, but it turns out to be inferior to a pointer trie. Let's start with a closed hash trie, along the lines suggested by Welch. To support V.42bis-style incremental updates, we must add an 8-bit count field to each hash-trie node, to keep track of the number of its descendants. A leaf node has a zero count; whenever a leaf is deleted, its parent's count field should be decremented. Total storage per entry in the hash table would then be 6 bytes: a 1-byte suffix character, a 2-byte pointer to the prefix (its parent node), a 1-byte count field, and a 2-byte code number. If the hash table is 90 percent full, we have 6.6 bytes per trie node, making this structure slightly larger than the 6 bytes per node in a minimal pointer-list trie. Also, the decompressor requires another 2 bytes per dictionary entry to maintain a mapping from codes onto hash indices. Closed hash tries are therefore at a significant disadvantage in the present memory-limited implementations of V.42bis.

If 64K-node dictionaries ever became prevalent, a random or otherwise incompressible input file would expose a weakness in a pointer-list trie. The average search path would be 128 characters long (approximately), as opposed to a small constant in a hash trie. The worst-case search path in the pointer-list trie would only be about 256 characters, however, as opposed to 64K in the hash trie. (V.42bis, unlike Welch's algorithm, will not limit the length of a collision chain in a hash search.) I conclude that pointer-list tries will be a preferred implementation for V.42bis, due to better worst-case performance, even on the largest dictionaries.

sent to the computer. The computer's response to a keystroke is variable: Sometimes no data is sent, sometimes a full-screen update of 2,000 characters is transmitted. Ideally, the computer's response to a user's keystroke would be displayed on the user's terminal within 0.1 second or so.

The desired peak bandwidth is thus about 2,000 characters in 0.1 second, or 20,000 cps, from the computer to the terminal. Much less data flows in the other direction, since few users type more than 10 characters per second. However, a terminal with programmable function keys can transmit 100 or more characters as a result of a single-user keystroke, for a peak (desired) bandwidth exceeding 1,000 cps.

I see no sign that these high-bandwidth standards are displacing the ubiquitous EIA-232 interface, nor have I seen any terminals with high-speed built-in modems.

Host computer bandwidth. When characters are sent from the computer to the terminal, an application program, such as an editor, running on that computer typically generates the characters one at a time. The application program may call an operating system routine to transmit each character. Alternatively, it may make one operating system call for each line of 64 characters, or possibly just one call per screenful of about 25 * 80 = 2,000 characters. Obviously, the character-by-character processing in the application software, and in the operating system, is a potential communication bottleneck. If the application is complicated, if the computer is not very fast, or if the computer is multiprocessing many applications simultaneously, it may be unable to support the 20,000-cps bandwidth peaks desired by the user.

Computer-to-modem bandwidth. The next step in transmitting data from the computer to the user's terminal is for the operating system to send the character, or block of characters, to a modem attached to the computer. This modem then transmits the characters to the modem connected to the user's terminal, which then sends the characters to the user's terminal. Finally, the user's terminal displays the characters. Each of these steps is potentially a bottleneck. For example, there may be a low-bandwidth connection between the computer and its modem, a low-bandwidth modem-to-modem connection, or a low-bandwidth modem-to-terminal connection. And the user's terminal may be incapable of accepting and displaying characters as rapidly as they are sent.

At my branch of the University of Minnesota, our campus mainframes and workstations are networked to a bank of modems. This computer-to-modem link has an upper limit of about 1,000 cps. One reason for this limit is applicable to many personal-computer-to-modem connections as well: Every character transmission causes an interruption of the computer's CPU. Most computers cannot handle more than about 1,000 interrupts per second.

Modem-to-modem bandwidth. The modem-to-modem connection is the easiest to specify and analyze. A V.42bis modem can send about 1,060 cps of uncompressed data on a typical telephone connection. If 2.3:1 compression is achieved, as is typical on English text, the available bandwidth rises to about 2,440 cps. However, if the telephone connection is impaired (but still within regulatory limits), the bandwidth will drop by 30 percent or more,[7] or about 1,600 cps for text compressible at 2.3:1.

Modem-to-terminal bandwidth. This bandwidth will present a bottleneck in some cases. Typically, the interface is an EIA-232-D, which was originally designed to support at most 1,920 cps. Note that this is significantly lower than the 2,440-cps bandwidth of a V.42bis modem-to-modem connection when English text is transmitted. Fortunately, most V.42bis modems, and some high-speed terminals and computers, can communicate reliably at 3,840 cps, or more, over an EIA-232-D interface.

The EIA-232-D standard is a recent update of the venerable RS-232C convention, which itself has been updated several times. Yet another revision is in the works, but the document describing EIA-232-E was unavailable at press time.

Another way to obtain a higher modem-to-terminal bandwidth would be to avoid the EIA-232-D interface between terminal and the modem. One could build the modem into the terminal itself, in much the same way as an internal modem is attached to a bus in a personal computer. Alternatively, one could use a high-speed terminal-modem interface such as the 10-Mbaud EIA-422-A or the 100-Kbaud EIA-423-A. However, I see no sign that these high-bandwidth standards are displacing the ubiquitous EIA-232 interface, nor have I seen any terminals with high-speed built-in modems.

Terminal display bandwidth; flow control. Most modern terminals can receive and transmit short bursts of information at 3,840 cps but are unable to maintain this bandwidth for an extended period of time. Then, the following flow control mechanism is invoked to avoid losing data. When a terminal's input buffers become nearly full, it sends a pause signal to the modem. The modem's data buffers will then start to fill up, eventually causing it to send a pause signal to the transmitting modem. The transmitting modem's output buffers will then start filling up, possibly causing it in turn to send a flow control signal to the transmitting computer.

A similar set of signals enables the transmission of data to be reinitiated between any pair of components in the path between the application program and the user's terminal. If the flow control mechanism has low latency, or if it is not invoked very often, this mechanism permits the transmission to proceed at the minimum bandwidth between any pair of components.

Performance gain from a V.42bis modem. As may be gathered from the foregoing discussion, it is not easy to know if the data compression feature of the V.42bis modem will result in a perceptible increase in performance over a noncompressing V.42 modem. V.42bis offers higher bandwidth for compressible data than does V.42, but unless the telephone connection is the bottleneck, there will be no increase in end-to-end bandwidth.

My experience may not be atypical: I have never observed more than 1,000-cps bandwidth on full-screen updates of a text editor on my home terminal (a Wyse-185) when it is connected with a V.42bis modem to any of the computers at my university. Since this bandwidth is below the 1,060 cps provided by V.42, I see essentially no benefit from V.42bis. One problem is that the connection from my workstation to my university's modem bank has a low bandwidth. I could sidestep this connection by installing my own modem on one of my workstations, or I could ask for an improvement in my workstation's connection to the modem bank, but frankly, this seems like more trouble than it is worth. I can tolerate a 1-second display time for a 1,000-character screen of text, and I am not willing to spend very much time or effort in improving this to 0.3 or 0.5 seconds.

However, some users have seen notable performance gains from V.42bis. High-speed personal computers with high-performance EIA-232-D ports exchange highly compressible (for example, spreadsheet) data at speeds approaching the 3,840-cps limit of the EIA-232-D interface.[7] This nearly fourfold improvement over the 1,060-cps limit of a V.42 modem yields sizable cost savings when transmitting long files over a long-distance telephone connection. Even over a local telephone connection, there would be sizable equipment and personnel savings in any application that involved transmitting long, compressible files. A megabyte of highly compressible data might be transmitted in as little as four minutes over a V.42bis connection, instead of 15 minutes on a V.42 connection.

Data-compression software. From the preceding paragraph, one might assume that any application involving computer-to-computer communication of compressible files will benefit from V.42bis. This is not the case. The transmitting computer could run a compression algorithm on its data before sending it to a noncompressing V.42 modem; the receiving computer can run the corresponding decompression algorithm. This approach permits the user to select the compression method most suitable for the data being transmitted. If a user transmits scanned image data, for example, bypassing the V.42bis feature helps more (10:1 compression is routine). Another advantage of compressing data before transmitting it to the modem is that an EIA-232-D connection is no longer a bottleneck: It is perfectly capable of supporting the 1,060-cps bandwidth of a noncompressing V.42 modem.

There are two disadvantages to this software data compression scheme. Someone must install and maintain appropriate software on both computers. In addition, the compression software will compete with the application software for CPU time (and memory space) on both computers.

Each manufacturer must recoup its engineering costs, marketing costs, and approximately $40,000 in V.42bis patent licensing fees.

The V.42bis market. Due to the factors just discussed, the market for V.42bis modems is limited. In computer-to-computer applications, software data compression might offer better performance. In computer-to-terminal applications, the modem might not be the bottleneck. Thus there should not be a large price differential between V.42 and V.42bis modems. Indeed, one of the challenges to the V.42bis standards definition team was to develop a very low-cost data compression method.

In mid-1990, shortly after the first V.42bis modems hit the market, a typical V.42 modem listed for $800 or more, but could be purchased for just over $600. Additionally, about $50 was charged for the V.42bis feature. In the case of the MultiTech product line, this feature was a minor modification to an existing V.42 modem. The modification involved adding another 8-Kbyte RAM chip and adding code to the ROM for the data compression software. The existing Z80 microprocessor apparently handled the extra CPU load of running the V.42bis data compression algorithm in addition to the V.42 error-correction protocol. The manufacturing cost of adding V.42bis to an existing V.42 modem was thus a few dollars. The profit margin on V.42bis might seem enormous to some readers, however each manufacturer must recoup its engineering costs, marketing costs, and approximately $40,000 in V.42bis patent-licensing fees.

In a completely redesigned modem, the V.42bis feature is extremely low in production cost. Such a modem would have a 16- or 32-Kbyte RAM chip instead of the 8-Kbyte RAM chip in a V.42 modem. It might also have a slightly larger ROM to accommodate the data compression software.

It is thus easy to understand why there were so few high-speed modems on the market in late 1991 that do *not* include the V.42bis feature. At worst (on incompressible data or on bandwidth-limited systems), V.42bis doesn't help; at best, it offers almost a fourfold improvement in bandwidth at a nominal increase in cost.

It will soon become rare to buy a high-speed modem without BCH error control.

V.32bis modems. The higher bandwidth V.32bis was recently approved by the CCITT, improving the raw bandwidth of a V.32 modem-to-modem telephone connection by 50 percent from 9,600 bps to 14,400 bps. Assuming the overhead of V.42 error correction remains at about 12 percent for this higher bandwidth connection, a V.32bis modem with V.42 error correction has a throughput limit of about (14,400/8)/112% = 1,610 cps on uncompressed data. This bandwidth limit would rise significantly to 3,700 cps on data that is compressible at 2.3:1 when the V.42bis data compression feature is used.

Note that a V.32bis modem has no advantage over a V.42bis (V.32) modem whenever the V.32 modem-to-modem connection is not a bottleneck. This case is likely to occur whenever highly compressible data is being transmitted.

Measured bandwidths on real systems. No modem will run at its limiting bandwidth in a real system, with its nonideal flow control, limited buffer sizes, and slow EIA-232-D interfaces. For this reason, a number of researchers have constructed systems consisting of two PCs, two modems, a real or simulated telephone connection, and a collection of test files. As I discovered recently, the convention in this field is to measure bandwidths, in bits per second (bps), at the EIA-232-D interface.

For example, Glass measured throughputs of 22,000 bps on a text file with a V.32bis/V.42bis modem, and 16,000 bps with a V.32/V.42bis modem.[9] Since 10 bit-times on the EIA-232-D interface are required to send an 8-bit character, Glass's measurement translates into *data* throughputs of 2,200 cps and 1,600 cps. These throughputs are about 40 percent lower than the ones I calculate here for text-file transmissions on these modems, leading me to conclude that Glass's system is bandwidth-limited somewhere other than in the modem-to-modem connection. My conclusion is supported by the fact that Glass measured only 24,000 bps (2,400 cps) on highly compressible database files, no matter which modem he used. This is only 63 percent of the bandwidth limit of his EIA-232-D connection which, ideally, would run at full bandwidth on a highly compressible file transmission.

Using a different test system, Henderson measured 24,000-bps throughput on a text file transmitted on a V.32/V.42bis modem.[10] This data throughput of 2,400 cps very closely agrees with the 2,440-cps bandwidth limit for V.32/V.42bis text transmissions (assuming 2.3:1 compressibility and 12 percent overhead for error correction). I conclude that Henderson's system does not suffer from bandwidth bottlenecks, other than the modem-to-modem connection under test.

The future. Another development in telephone network signaling, hovering on the near-future horizon for many years, is the Integrated Services Digital Network (ISDN). When and if local telephone loops are digitized, dial-up connections will run at 8,000 cps, without a modem. This is over twice the bandwidth of EIA-232-D, so this standard will finally be obsolete. One might make ISDN file-transfer connections with a special interface board on a personal computer, implementing V.42bis data compression and V.42 error correction. The result would be an effective bandwidth of nearly 20,000 cps for English text and 13 Kbytes/s for executables.

Even though digital signaling (ISDN) may never become widely available, other methods of analog signaling on the telephone network may become standardized soon. The 14,400-bps signaling of V.32bis has been surpassed by V.fast transmissions at 19,200 bps, and even 24,000 bps. Standardization of V.fast is expected soon. In conjunction with V.42bis data compression, data throughputs of up to 7,000 cps for English text are thus feasible. The slow EIA-232-D computer-to-modem interface becomes even more problematical at these data rates.

As computational power and memory become cheaper, more powerful methods for data compression and error control will no doubt become commonplace in the modem market. V.42bis could be revised to take advantage of recent work on data compression algorithms. Also, the V.42 error control protocol could be revised to incorporate a BCH or Reed-Solomon code that would greatly reduce the bandwidth and latency problems associated with V.42's retransmission after each single-bit error. One referee asserts that a BCH encoder/decoder chip would cost just $50 at present, if purchased in quantity. With the price multipliers typical of a high-technology field, this would increase the cost of a modem by perhaps $200. However, the cost of a BCH chip will no doubt drop rapidly in the next few years, possibly to the point where it will soon become rare to buy a high-speed modem without BCH error control.

Experimental results

John Copeland, of the V.42bis study group, kindly sent me C-language source code for an implementation of the V.42bis standard. The comments and experimental results in this section are obtained from his version 6.83, dated July 16, 1989, of the hv42b3.c code. Hayes Microcomputer Products, Inc. owns the copyright to this code, which it says is "distributed for experimentation aimed at developing a CCITT V.42 data compression standard."

The Hayes code implements the V.42bis dictionaries as rchild/lsibling/rsibling/parent pointer tries. It sorts sibling lists by their suffix character.

When I ran the Hayes code on my Sun 3/50. it compressed my 100-Kbyte test files at rates of 6 to 10 Kbytes of input data per second. On a 10-MHz Z80, I estimate that the Hayes code would run two to four times slower, due to the lower MIPS rate and the narrower data bus. Some optimization of the code may thus be necessary to keep the CPU from becoming the bottleneck in a V.42bis modem, especially since the CPU in such a modem has a number of other tasks. The CPU must handle V.42bis compression on the transmission channel, V.42bis decompression on the receiving channel, as well as V.42 error correction and V.32 packetizing on both channels. (Perhaps the ATI 9600etc/e modem has a CPU bottleneck of this sort, which would explain why it was unable to deliver more than 1,977-cps throughput, on an experimental setup where its fastest competition ran at 3,440 cps.)[7]

I have run a number of compression measurements on the V.42bis implementation used by the study group. All measurements reported involve the following five files. The first, labeled csh, is the 122,880-byte executable C shell in my Sun 4.2 Unix. The second test file, Essays, consists of 176,369 bytes of concatenated freshman essays. It is cleartext English, containing no text-formatting commands. The third file, News, is 210,931 bytes of articles representing a (small) portion of a day's Usenet feed. This file is a good test of a compression algorithm's ability to respond quickly to a rapidly changing source, since the article header fields are quite different from the article bodies, and the article bodies are also variable (ranging from cleartext English to source code). The fourth file, Pascal, is 119,779 bytes of source code to a student-written program and is highly compressible, due to its repeated use of long identifiers. The fifth and final test file, Tex, is the 201,746-byte Latex manual, vintage 1984, containing both cleartext and text-formatting commands.

Table 1 shows the compression ratios obtained by the Hayes code on the five test files, when the length of the longest code word is 16 and the maximum length of the dictionary varied between 512 and 4,096. In this table, and throughout this section, I define compression ratio to be the length of the compressed file divided by the length of the input file. A compression ratio of 0.43 is thus the 2.3:1 compression I've already mentioned as the performance of V.42bis on a sample text file, Essays.

Note that increasing the size of the dictionary noticeably affects the compression ratio for all files save csh. Perhaps the most striking feature of Table 1, however, is that even a short,

Table 1. Effect of dictionary size on V.42bis compression ratio, when maximum code word length is 16.

	Dictionary size			
Test files	512	1,024	2,048	4,096
Csh	0.65	0.62	0.62	0.62
Essays	0.57	0.49	0.45	0.43
News	0.70	0.64	0.60	0.56
Pascal	0.52	0.41	0.35	0.31
Tex	0.59	0.50	0.45	0.42

Table 2. V.42bis compression ratio, for various dictionary sizes, when maximum code word length is 6.

	Dictionary size			
Test files	512	1,024	2,048	4,096
Csh	0.65	0.63	0.63	0.64
Essays	0.57	0.49	0.46	0.43
News	0.70	0.64	0.60	0.57
Pascal	0.53	0.43	0.38	0.35
Tex	0.60	0.51	0.46	0.44

512-code word dictionary is sufficient to obtain good compression on the test files. I've presented results only for test files long enough that large dictionaries may become useful. On short files, short dictionaries do just as well as long dictionaries.

Table 2 is identical to Table 1, except that the length of the longest code word is restricted to 6 characters. These figures are only slightly larger than those of Table 1, indicating that the code word-length parameter is not very important to V.42bis performance, at least for the five test files.

Table 3 on the next page shows the compression ratios obtained by the Unix Compress utility[2] on the five test files, as a function of dictionary size. Compress is a poor algorithm for small dictionaries. Even with a 64K dictionary, it is only slightly better than V.42bis with a 4K dictionary.

Tables 4 and 5 compare the performance of the Hayes code with a 4K-code word dictionary (maximum code word length is 16) with that of a number of other Ziv-Lempel variants. Where possible, I use the abbreviations of the Fiala-Greene work.

Table 4 consists of those algorithms which, like V.42bis, send code word indices in straight binary form. The algorithms of Table 5 gain compression, at some penalty in speed, by using Huffman or arithmetic codes to compress the code

Table 3. Unix-Welch compression ratio, for various dictionary sizes.

Test files	Dictionary size 512	1,024	2,048	4,096	65,536
Csh	0.98	0.82	0.89	0.75	0.64
Essays	0.67	0.53	0.49	0.46	0.40
News	0.91	0.73	0.71	0.67	0.54
Pascal	0.70	0.50	0.43	0.37	0.30
Tex	0.85	0.61	0.53	0.48	0.39

word indices.

Algorithms under text. Expert readers may be interested in the following paragraphs; others may skip to the Discussion section.

LZRW1 is a high-speed ZL1 scheme[11] recently released into the public domain by Ross Williams.[12]

Allb is a very high-speed ZL2 scheme[1] released in late 1990 into the public domain for research use only, by Brandon S. Allbery. This code can be retrieved under the name compact_sv, from volume 15, number 89, of the archives for the Internet news group, comp.sources.misc.

UW is the standard Unix-Welch Berkeley Compress utility, a ZL2 variant,[1] with 16-bit codes (64K dictionary entries).[2]

A1, B1, A2, B2, and C2 are Fiala and Greene's algorithms. A1 is a ZL1 scheme[11] with a 4K-character buffer, augmented with a literal transmission mode using a suffix trie to accelerate the buffer searches. Its compressed output contains two types of code words: literal *x*, meaning that the decompressor should pass the next *x* characters directly to the output, and copy *x y*, meaning that the decompressor should go back *y* characters in the output buffer and append the next *x* characters to the output buffer. B1 is similar to A1, but optimized for speed. It builds a Patricia trie with only some of the substrings contained in the last 4K characters of input. A2 and B2 are similar to A1 and B1 with 16-Kbyte buffers and a static Huffman back end. C2 is designed for maximum compression. Its Patricia trie contains all the substrings contained in the last 16K characters of input. It gains compression efficiency over the usual ZL1 scheme, since it does not waste code space on duplicate substrings. Because A2, B2, and C2 use a Huffman code to send trie indices, I have listed their compression ratios in Table 5.

MW1 and MW2 are the Miller-Wegman algorithms,[3] as implemented by Dan Greene of Xerox PARC.[1] MW1 is a ZL2 variant, with a 4,096-entry dictionary, a least recently used replacement algorithm, and Welch-style output coding. MW2 is like MW1, augmented with string extension. It is thus an ID-LRU, in Storer's terminology.[5]

MWP, of Table 5, is a high-compression, high-speed implementation of MW2 by Roberto Pasqui of IBM Italy. It uses a static Huffman code with only three code lengths. The shortest codes refer to the most recently used set of 128 code words; medium-length codes refer to a set of less recently used 512 code words; and the longest codes refer to the oldest 4,096 code words.

Y64 and Y512 are the *yabba* coders, placed into the public domain recently by Dan Bernstein. Readers can retrieve these codes from volume 24, postings number 73 through 76, of the comp.sources.unix archives. *yabba* is a ZL2 variant[1] that adds one string *pc* to its dictionary for each input character *c*, where *p* is the longest suffix of the already processed input such that *p* is currently in the dictionary. The higher compressing Y512 has a 512K-entry dictionary; Y64 has a 64K-entry dictionary.

Like MWP, the Rogers-Thomborson code RT[13] uses a back-end coder to transmit frequently used code words with fewer bits than the rarely used ones. However, since RT uses an adaptive arithmetic code rather than a static Huffman code, it is very slow. RT grows its dictionary in a similar fashion to the MW2 and MWP algorithms, except that it does not add new dictionary entries ending with a blank space when extending characters. In such cases, only string extension oc-

Table 4. Compression ratios of Ziv-Lempel algorithms that do not also employ Huffman or arithmetic coding. Boldface entries are row minima.

Test files	Algorithms V42	LZRW1	Allb	UW	A1	B1	MW1	MW2	Y64	Y512
Csh	0.62	0.65	0.62	0.64	0.61	0.62	0.62	**0.57**	0.64	0.60
Essays	0.43	0.59	0.48	0.40	0.45	0.44	0.42	0.40	0.39	**0.37**
News	0.56	0.65	0.60	0.54	0.56	0.55	0.56	0.53	0.54	**0.49**
Pascal	0.31	0.39	0.35	0.30	0.25	0.25	0.31	**0.17**	0.25	0.22
Tex	0.42	0.52	0.46	0.39	0.41	0.39	0.41	**0.35**	0.40	0.35

curs. This heuristic tends to increase the fraction of dictionary entries that are complete English words or phrases, when the input is textual. Since there are usually few ASCII blanks in a binary file, their compression is not degraded much. Note that RT compresses slightly better than MWP, on all but the csh binary. Most of this improvement is due to the nonblank heuristic, not to the arithmetic coding; but this hypothesis deserves test.

Table 5. Compression ratios of algorithms employing Huffman or arithmetic coding. Boldface entries are row minima over Tables 4 and 5.

	Algorithms								
Test files	A2	B2	C2	MWP	RT	LHarc	Freeze	BSTW	ppmC
Csh	0.54	0.55	0.52	0.51	0.55	0.51	0.50	0.69	**0.47**
Essays	0.40	0.39	0.36	0.36	0.34	0.40	0.39	0.39	**0.29**
News	0.48	0.48	0.44	0.47	0.45	0.48	0.46	0.53	**0.40**
Pascal	0.15	0.14	**0.13**	0.17	0.16	0.19	0.15	0.24	0.18
Tex	0.34	0.33	0.31	0.32	0.31	0.36	0.33	0.38	**0.27**

LHarc (see Table 5) is a public-domain code from Japan in common use for distribution of personal computer software on floppy disks. To install such packages, users must typically wait minutes for the software to be decompressed onto their hard disk or alternate floppy drive. I obtained a Unix version of LHarc over the Internet, in volume 11, numbers 17-18, of the comp.sources.misc archives. On my Sun 3/50, LHarc runs at about 2 Kbytes/s, or about 15 times slower than the Unix Compress utility UW. As may be seen from Table 5, however, LHarc has better compression ratios. No documentation was included with my version of LHarc. It appears to run in two passes: a ZL1 coding[11] with a 4-Kbyte buffer followed by a dynamic Huffman recoding of the Ziv-Lempel indices. This Huffman back end probably accounts for most of the compression improvement and speed degradation, relative to UW.

Freeze is a public-domain code that recently emerged from the Soviet Union. It is archived in volume 17, numbers 68 and 74, of the comp.sources.misc news group. Like LHarc, Freeze is a ZL1 (first Ziv-Lempel) variant with a dynamic Huffman back end, although it has an 8-Kbyte buffer. Since single-character literals appear in the output stream, Freeze is more precisely an LZSS (Lempel-Ziv-Szymanski-Storer)[14] variant.[1,12] Its output coding appears to work a little better than that of LHarc, and it runs at about the same speed. Of a 512-symbol alphabet, the first 256 symbols denote single-character literals; one symbol marks an end-of-file; and the last 255 symbols indicate the length l of the match ($1 < l \leq 256$). If a multiple-character match is selected, the next bits in the compressed stream indicate the most-significant 6 bits of the position of the match. These bits are sent with a static Huffman code of 1 to 8 bits. The 7 least significant bits of the match position transmit in binary.

As may be seen in Table 5, the Bentley-Sleator-Tarjan-Wei code[15] achieves very good compression ratios for every file except csh. Dan Greene's implementation provided the data in Table 5. This data agrees with my independently coded implementation to within 0.01.

BSTW's compression ratios are remarkable, it uses less than 6 Kbytes of space for its data structures. A BSTW compressor, in my implementation, uses two linked lists of 239 cells containing words of up to 16 characters each. One list contains only alphanumeric words; the other list contains nonalphanumeric words. The compressor parses the input file into an alternating sequence of maximal length alphanumeric and nonalphanumeric words. Zero-length words in the lists allow for the eventuality that the input contains alphanumeric (or nonalphanumeric) substrings of length greater than 16. A previously encountered word transmits with an adaptive Huffman encoding of the list position and then moves to the front of its list. A new word transmits with an escape code followed by an adaptive Huffman encoding of the cleartext word.

The Bell-Cleary-Moffat-Witten code ppmC[16,17] of Table 5 is based on a variable-order Markov modeling of the source, with arithmetic coding used to communicate state transitions. It achieves impressive compression ratios, however it requires the largest amount of data space (730 Kbytes) of any algorithm in my tables.

Readers interested in more data on the relative performance of compression algorithms should access the comp.compression news group on Internet. One contributor, Peter Claus Gutman, is currently developing a large body of data on the Calgary corpus of test files.[17]

Discussion. As indicated in Tables 4 and 5, V.42bis obtains markedly worse compression ratios than many other algorithms. Let's look at competing algorithms to see if they offer a significant advantage in the V.42bis application.

The LZRW1 and AllB algorithms offer significantly worse compression ratios than V.42bis. However, they run two or three times more rapidly on my Sun 3/50 than any other algorithm I have tested. I observed speeds of 50-70 Kbytes/s and 90-120 Kbytes/s on this 1-MIPS machine, using Sun's optimizing compiler on these C codes. Even higher bandwidths could be obtained with CPU-specific code tuning. The LZRW1 and AllB algorithms would thus be preferable to V.42bis in applications that are cost-limited but not I/O chan-

Readers interested in more data on the relative performance of compression algorithms should access the comp.compression news group on Internet.

nel-limited, for example, in a compressing interface to a low-cost tape or disk drive. For the low-cost modem application, however, the better compressing V.42bis algorithm is clearly preferable, since a 10-MHz Z80 is sufficient to keep up with a telephone modem's low-speed serial I/O channels.

The UW algorithm achieves slightly better compression than V.42bis on the test files, at the expense of 16 times the data space. Since a low-cost modem is limited in memory, UW is not an appropriate choice. Furthermore, as shown in Table 3, a memory-limited UW is not competitive in compression ratio with V.42bis.

Fiala and Greene's algorithm A1 requires 145 Kbytes to compress with a 16K-entry dictionary; their other algorithms require even more space. Further research is necessary to decide if any of these algorithms are competitive with V.42bis, when restricted to a 10- or 20-Kbyte dictionary and an 8-bit microprocessor. Still, the A2, B2, and C2 algorithms would be a promising starting point for research on a future data compression standard.

Despite its LRU-Huffman back end, MWP runs at a respectable 10-20 Kbytes/s on my test files, on an IBM PS/2 mod 80, 16-MHz 80386 under DOS 4.0, according to private correspondence from Victor Miller. By way of comparison, the Unix Compress utility UW runs at 25-50 Kbytes/s on the test files on my Sun 3/50, and my V.42bis implementation runs at only 6-10 Kbytes/s. Even though my V.42bis implementation is not optimized for speed, I believe MWP would be at least as fast as V.42bis, even on an 8-bit microprocessor such as a Z80. Despite its impressive speed and compression ratios, however, MWP is not immediately applicable to the design of today's low-cost V.42bis modems, since it requires 60 Kbytes of space for its data structures.

BSTW requires just 6 Kbytes of data space, so it would fit in a low-cost modem. However, its adaptive Huffman encoding would severely limit the bandwidth of any such modem's low-performance processor. A much more expensive modem could employ a special-purpose, high-bandwidth, Huffman- (or arithmetic-) encoder/decoder chip, as well as a high-speed hardware implementation of BSTW.[18] The poor performance of BSTW on nontextual data could pose a problem in many applications, although one might envision a revision to V.42bis allowing a compressor to enter BSTW mode whenever suitable data was received.

As noted earlier, a V.32bis/V.42bis modem is usually limited by factors other than its compression ratio. In high-performance, relatively cost-insensitive applications, bottlenecks such as the EIA-232-D can be removed, and other data compression schemes would be markedly superior to V.42bis. Thus MWP (perhaps with the RT conditional dictionary entry heuristic), the Fiala-Greene algorithm B2, BSTW, and ppmC deserve attention in the next round of standards-making activity.

THIS EXAMINATION OF SOME OF THE PRACTICAL, algorithmic, and marketing aspects of the V.42bis standard has not uncovered any serious flaws in its specification. On the contrary, the algorithm embodied in this standard is competitive with other Ziv-Lempel variants, if these are required to run in a small data space on an 8-bit microcomputer.

My data shows that a well-implemented V.42bis modem can achieve 2.3:1 compression on English text, 1.6:1 compression on 68000 object code, and 3.2:1 compression on Pascal source code. The best compression ratios I observed, for any data compression algorithm, were 3.4:1 on English text, 2:1 on object code, and 7:1 on my highly compressible sample of Pascal. The algorithms achieving these ratios require significantly more computational resources than does V.42bis, so they would not be viable in the current market for modems. Still, it is clear that a revision to the V.42bis standard could offer 50 percent or more additional bandwidth in many applications.

The future of data-compressing modems is uncertain. Microcomputers and their memory are rapidly becoming cheaper, by 20 percent to 30 percent per year. Thus the quick-and-inexpensive algorithmic choices embodied in V.42bis will soon be outmoded, and the more computationally expensive but better-compressing algorithms I've mentioned will become economical for text transmission. However, text transmission is only a fraction of current computer communication over the telephone network, and this fraction is rapidly decreasing. Source-specific compression algorithms (for voice, image, fax) can be run efficiently on a personal computer, achieving much higher compression ratios than are offered by any of the algorithms discussed here, for these nontextual sources.

Following this line of reasoning, the V.42bis standard will continue to be used in low-cost computer-to-ASCII-terminal communications, but it will find less and less applicability in computer-to-computer communications. I would not expect to see many modems making use of the V.42bis feature in the year 2000, when a 10-MIPS personal computer should be only a little more expensive than an ASCII terminal. Source-specific data compression algorithms will run on such computers at high bandwidth, not only for communication but also for mass storage devices.

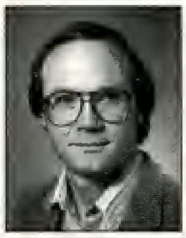
Acknowledgments

Former UMD students Clyde Rogers and Julie Redland, or Dan Burrows and Bill Marko of UMD Information Services, ran many of the experiments I've described. I am indebted to John Copeland, of Hayes Microcomputer Products, Inc., for a copy of the source code to a V.42bis-compliant compressor and for the test files used by the V.42bis study group. I thank Dan Greene of Xerox PARC, Tim Bell of the University of Calgary, and Victor Miller of IBM for taking the time and trouble to run our test files through their compressors. Telephone conversations with Ashok Patel and Warren Henderson corrected my misunderstandings about their throughput measurement data. Last but not least, I thank the (four!) referees of my manuscript for their detailed and thoughtful reviews.

The National Science Foundation, through its Design, Tools and Test Program, supported this research under grant number MIP 9023238.

References

1. J.Ziv and A.Lempel, "Compression of Individual Sequences via Variable-Rate Coding," *IEEE Trans. Information Theory*, Vol. 24, No. 5, 1978, pp. 530-536.
2. T.A. Welch, "A Technique for High-Performance Data Compression," *Computer*, Vol. 17, No. 6, June 1984, pp. 8-19.
3. V.S. Miller and M.N. Wegman, "Variations on a Theme by Ziv and Lempel," in A.Apostolico and Z.Galil, eds., *NATO ASI Series,* Vol. F12, Combinatorial Algorithms on Words, Springer-Verlag, Berlin, 1985, pp. 131-140.
4. J.A. Storer, "Textual Substitution Techniques for Data Compression," *ibid.,* pp. 111-129.
5. J.A. Storer, *Data Compression: Methods and Theory,* IEEE Computer Society Press, Los Alamitos, Calif., 1988.
6. E.W. Fiala and D.H. Greene, "Data Compression with Finite Windows," *Commun. ACM*, Vol. 32, No. 4, Apr. 1989, pp. 490-505.
7. M. Byrd, "9600-bps Modems: Breaking the Speed Barrier," *PC Magazine*, Dec. 11, 1990, pp. 307-346.
8. U. Black, *Physical Level Interfaces and Protocols*, CS Press, 1988.
9. B. Glass, "High Speed Modems," *PC World*, Dec. 1991, pp. 236-242.
10. W.L. Henderson, Jr., and S.S. King, "Testing V.32 Modems—the Right Way ... (cont'd)," *Data Communications*, May 1991, pp. 99-105.
11. J.Ziv and A.Lempel, "A Universal Algorithm for Sequential Data Compression," *IEEE Trans. Information Theory*, Vol. 23, No. 3, 1977, pp. 337-343.
12. R.N. Williams, "An Extremely Fast Ziv-Lempel Data Compression Algorithm," *Proc. Data Compression Conf.*, IEEE, New York, Apr. 1991, pp. 362-369.
13. C. Rogers and C.D. Thomborson, "Enhancements to Ziv-Lempel Data Compression," *Proc. 13th Int'l Computer Software and Applications Conf.*, Sept. 1989, pp. 324-330.
14, J.A. Storer and T.G. Szymanski, "Data Compression via Textual Substitution," *J. ACM*, Vol. 29, No. 4, 1982, pp. 928-951.
15. J.L. Bentley et al., "A Locally Adaptive Data Compression Scheme," *Commun. ACM*, Vol. 29, No. 4, Apr. 1986, pp. 320-330.
16. J.G. Cleary and I.H. Witten, "Data Compression Using Adaptive Coding and Partial String Matching," *IEEE Trans. Computers*, Vol. COM-32, No. 4, Apr. 1984, pp. 396-402.
17. T.C. Bell, J.G. Cleary, and I.H. Witten, *Text Compression*, Prentice Hall, Englewood Cliffs, N.J., 1990.
18. C.D. Thomborson and B. W-Y Wei, "Systolic Implementations of a Move-to-Front Text Compressor," *Proc. 1989 ACM Symp. Parallel Algorithms and Architectures*, ACM, New York, June 1989, pp. 283-291.



Clark Thomborson teaches computer science and computer engineering at the Duluth campus of the University of Minnesota but is currently on a one-year sabbatical at MIT, where he will teach and conduct research. He has also served on the faculty of the University of California, Berkeley, Computer Science Division.

Thomborson received his bachelor's degree in chemistry, master's in computer science/computer engineering from Stanford, and doctorate in computer science from Carnegie Mellon University. He has published more than 40 articles on special-purpose hardware implementations of algorithms, VLSI theory, graph theory, algorithmic analysis and the effects of military funding on academic science and engineering. He is a member of the IEEE and ACM.

Direct any questions regarding this article to the author at the Massachusetts Institute of Technology, Laboratory for Computer Science, 545 Technology Square, Cambridge, MA 02139; or e-mail at cthombor@theory.lcs.mit.edu.

Reader Interest Survey

Indicate your interest in this article by circling the appropriate number on the Reader Service Card.

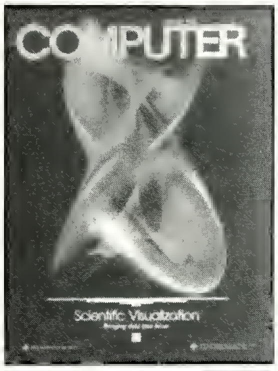
Low 165	Medium 166	High 167

IEEE COMPUTER SOCIETY

1992 Membership/Subscription Application

THE INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS, INC.

BENEFITS



Receive *Computer* automatically with membership. Written and refereed by experts, it features articles on the latest developments in computer technology, applications, and research, as well as survey and tutorial articles on topics covering the entire computer field.

Computer's popular departments include New Products, Product Reviews, Book Reviews, Standards, Open Channel (a reader forum), and Computer Society News.

Other membership benefits include the following:

- Discounts of up to 50% on **Computer Society Press books and videos**. Over 700 titles covering a broad spectrum of computer science topics including networking, communications, advanced systems, image processing, security, artificial intelligence, and visualization
- Discounts on registration to over 100 conferences, workshops, and tutorials each year
- Low subscription rates on special-interest periodicals
- Opportunity to participate in any of our 32 technical committees—networks of professionals with common interests in computer hardware, software, and applications
- Opportunity to participate in the development of more than 200 standards projects sponsored by the society
- Low member rates on COMPMAIL, the electronic mail service especially designed for computer professionals

Schedule of Fees

To join: see item 1, 2, or 3.
To subscribe: see item 4.

Membership dues and periodical subscriptions are annualized to, and expire on, December 31. Pay full- or half-year rate depending on date of receipt by the Computer Society as indicated below.

PRICES EXPIRE 12/31/92

		Half Year Mar 1-Aug 31	Full Year Sept 1-Feb 28
1	I don't belong to the IEEE and I want to join just the Computer Society	☐ $27	☐ $ 54
2	I want to join both the Computer Society and the IEEE*		
	I reside in Region 1-6 (United States)	☐ $58.50	☐ $117
	I reside in Region 7 (Canada)	☐ $53.50	☐ $107
	I reside in Region 8 (Europe, Africa, or the Middle East)	☐ $53.00	☐ $106
	I reside in Region 9 (Latin America)	☐ $49.50	☐ $ 99
	I reside in Region 10 (Asia and Pacific)	☐ $48.50	☐ $ 97

*ACM members who join both IEEE and the Computer Society may deduct $5 off the full-year rate; $2.50 off the half-year rate.

		Half Year	Full Year
3	I already belong to the IEEE and I want to join the Computer Society	☐ $11.00	☐ $22

IEEE Member Number ____________

4 SPECIAL-INTEREST PERIODICALS for new or current members

	issues per year	Half Year	Full Year
IEEE Computer Graphics and Applications	6	☐ $12.50	☐ $ 25
IEEE Design and Test	4	☐ $11.00	☐ $ 22
IEEE Expert	6	☐ $10.00	☐ $ 20
IEEE Micro	6	☐ $11.50	☐ $ 23
IEEE Software	6	☐ $12.50	☐ $ 25
Transactions on:			
Computers	12	☐ $12.00	☐ $ 24
Knowledge and Data Engineering	6	☐ $ 9.50	☐ $ 19
Parallel and Distributed Systems	6	☐ $ 9.50	☐ $ 19
Pattern Analysis and Machine Intelligence	12	☐ $12.00	☐ $ 24
Software Engineering	12	☐ $11.00	☐ $ 22
IEEE Annals of the History of Computing	4	☐ $ 8.00	☐ $ 16

☐ Visa ☐ MasterCard ☐ American Express

Charge Card Number (Minimum Charge $10)

Mo. Yr.
Exp. Date

Checks drawn in local currency on a bank in the country of origin accepted from members in Australia, Austria, Belgium, Canada, Denmark, France, Germany, Ireland, Italy, Luxembourg, Netherlands, Norway, Portugal, Spain, Sweden, Switzerland, UK, USA, and Yugoslavia.

Periodicals total $________

CA, DC residents add applicable sales tax $________

Membership fees $________

Total $________

CANADIAN residents add 7% GST; BELGIAN residents add 6% VAT $________

Total enclosed $________

I hereby make application for Computer Society membership and, if elected, will be governed by IEEE's and the society's constitutions, bylaws, and statements of policies and procedures.

MAILING ADDRESS

Full signature ________ Date ________

☐ Male ☐ Female

First name ________ Middle initial(s) ________ Last name ________ Date of birth ________

Street address ________ City ________ State/Country ________ Zip ________

EDUCATION (highest level completed) Course ________ Degrees received ________ Date ________

Name of educational institution ________

OCCUPATION Title or Position ________

Firm name ________

Firm address ________

City ________ State/Country ________ Zip ________

ENDORSER (an IEEE member, Senior Member, or Fellow)

Endorser's signature ________

Name (print in full) ________ IEEE Member No. ________

Street address ________

City ________ State/Country ________ Zip ________

Mail or fax to appropriate Computer Society office:

EUROPE: 13, Avenue de l'Aquilon, B-1200, Brussels, BELGIUM. Fax: 32-2-770-8505
PACIFIC RIM: Ooshima Building, 2-19-1 Minami-Aoyama, Minato-ku, Tokyo 107, JAPAN. Fax: 81-3-3408-3553
ALL OTHERS: 10662 Los Vaqueros Circle, PO Box 3014, Los Alamitos, CA 90720–1264 USA. Fax: 714-821-4010

Achieving Supercomputer Performance

for Neural Net Simulation
with an Array of Digital Signal Processors

Music, a DSP-based system with a parallel distributed-memory architecture, provides enormous computing power yet retains the flexibility of a general-purpose computer. Reaching a peak performance of 2.7 Gflops at a significantly lower cost, power consumption, and space requirement than conventional supercomputers, Music is well suited to computationally intensive applications such as neural network simulation.

Urs A. Muller

Bernhard Baumle

Peter Kohler

Anton Gunzinger

Walter Guggenbuhl

Swiss Federal Institute of Technology

Digital signal processing and scientific calculation often demand high computing power. Because many tasks in these applications have great potential for parallel processing, developers have proffered a number of parallel computer architectures. For example, the Warp[1] multiprocessor system has a ring architecture with a maximum of 10 processor elements. Its peak performance is 100 Mflops (million floating-point operations per second). A ring architecture also structures the Ring Array Processor,[2] which uses TMS320C30 digital signal processors as processor elements. Another parallel system based on DSPs is the DSP3,[3] which consists of multichip modules in a relatively complex communication network topology. The GF11[4] is a large SIMD (single-instruction, multiple-data) machine with a maximum of 566 processors and a peak performance of 11 Gflops. The Connection Machine CM-2[5] is a massively parallel computer with up to 65,000 simple processor elements arranged in a hypercube topology.

Such systems can outperform even conventional supercomputers. But a few parameters of parallel system design have a great impact on efficiency. At the Swiss Federal Institute of Technology, we are engaged in a project to identify these parameters by designing and building a parallel supercomputer system and demonstrating its performance in real-world applications.

Along with performance, price is an important factor of a computer system. Figure 1 (on the next page) shows that DSPs are in a good position for building systems with a high performance-to-cost ratio. Using an array of DSPs, a system can reach several Gflops at a significantly lower cost, power consumption, and space requirement than supercomputers of equal performance.

Our system, called Music (multi-signal-processor system with intelligent communication), is based on a parallel distributed-memory architecture and DSPs. A system with 45 processor elements and a peak performance of 2.7 Gflops is operational. The material cost of the Music system is US$60,000, corresponding to a theoretical selling price of approximately $300,000.

Music system hardware

Typically, in digital signal processing tasks, in many simulation applications, and in neural net simulations, an algorithm transforms a source data set into a destination data set. The amount of computing power required mainly depends on the data size, not on the complexity of the algorithm. Typical examples are filtering, convolution, matrix computations, and iteration steps in simu-

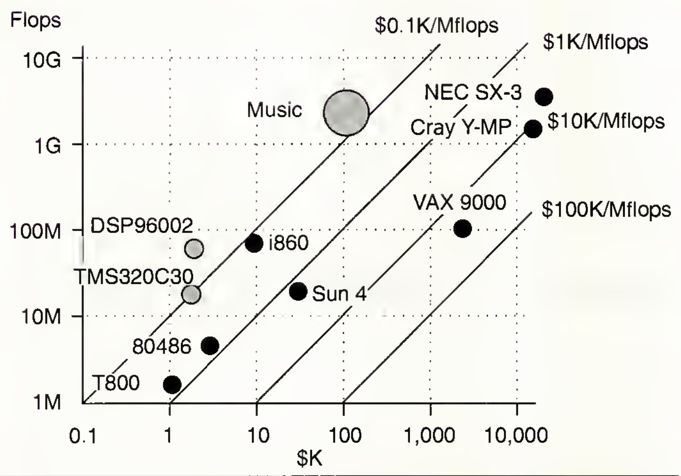


Figure 1. Cost and performance comparison of computer systems and processors. DSPs are indicated by gray dots. System costs include CPU and memory costs.

lation or finite-element computation. We can parallelize most of these algorithms efficiently in data space. That is, in an iteration step, each processor element computes a part of the output data, without communication to the other processors. Between steps a data redistribution takes place. Figure 2 illustrates a typical situation: At the beginning each processor element has a local copy of the complete input data set and computes a part of the output data set. The subsequent communication phase redistributes the data so that each processor element gets a copy of the complete produced data set or a new part of the produced data set, which then serves as input for the next step.

The key idea of the Music architecture is to support the collection and distribution of data sets among all processor elements by an independent communication network. The produced data subsets move among processor elements via a global ring bus, and the communication controller of each processor element copies the pertinent parts of the data into the processor element's memory. The hardware-implemented communication controllers allow communication in parallel with data computation. Figure 3 illustrates a typical timing situation. See the box for a simple performance model of this architecture.

Figure 4 on page 58 shows an overview of the Music system hardware. Each board contains three processor elements. Each processor element is composed of a DSP (Motorola 96002), up to 8 Mbytes of video RAM (VRAM), up to 1 Mbyte of static RAM (SRAM), and a communication controller logic cell array (Xilinx LCA XC3090). Because the communication network uses the VRAM's video interface for data communication, its activity affects normal data processing on the DSPs only slightly. Each board also contains a manager (Inmos T800 transputer), connected to the host interface of the DSPs. It is responsible for data and code uploading and downloading, time measurements, and dynamic adaption of processor loads. Figure 5 is a photograph of a Music board.

The managers of different boards interconnect by their transputer links, forming a standard transputer network. For the fast data throughput required by applications such as real-time image processing, we can add special I/O boards. A commercially available transputer module acts as host processor. It has access to mass storage and user terminals and is responsible for managing the complete Music system from the user's point of view.

The communication network is a pipelined ring bus, operating at a 5-MHz clock rate. Its width is 40 bits: 32 data bits and 8 token bits. The tokens contain the identification of the processor element that produced a particular data value. On each clock cycle, the processor elements shift a data value to their right neighbors and receive a new value from their

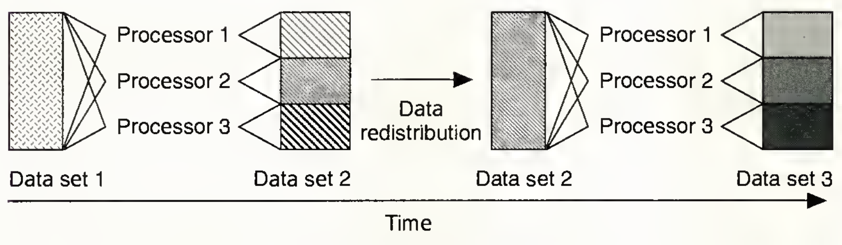


Figure 2. Two iteration steps of a data-parallel algorithm.

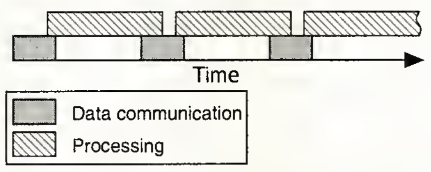


Figure 3. Representative timing diagram of Music system activities. Communication and processing are independent and can therefore overlap in time.

Performance measurement and speedup model

The most important performance measure in scientific applications is computing time. Since computing time greatly depends on the task, we often measure performance as the number of floating-point operations a system can execute in one second (flops or Mflops) instead. This measure usually means the peak performance for a function optimally adapted to the hardware architecture.

The purpose of parallel computers is to increase performance by using several processors working in parallel. The speedup factor of a parallel system denotes how many times faster a certain algorithm is carried out on a system containing n processor elements than on a system with one processor element.

How much speedup does the Music architecture achieve? An important parameter is the ratio between communication time t_{comm} and computation time t_{comp} for a data set on a single processor element. This ratio is known as the complexity factor c:

$$c = \frac{t_{\text{comp}}}{t_{\text{comm}}}$$

If communication and computation do not overlap and the computation speed scales linearly with the number of processor elements, the total processing time is

$$t_{\text{tot}}(n) = \frac{t_{\text{comp}}}{n} + t_{\text{comm}}$$

The latter assumption is not always true, as we know from Amdahl's law.[1] It is, however, a good approximation if the data sets are large with respect to n. The speedup factor s then is

$$s(n) = \frac{t_{\text{comp}}}{t_{\text{tot}}(n)} = \frac{n \cdot c}{n + c} \qquad \textbf{(A)}$$

Because it increases when communication and computation overlap, Equation A can be taken as a lower bound. Notice that the maximum speedup in Equation A for a large number of processor elements is

$$\lim_{n \to \infty} s(n) = c$$

The diagram in Figure A illustrates the lower bound of speedup for various complexity factors. A good speedup occurs if the complexity factor is high with respect to the number of processor elements. Very simple programs running on Music, such as a 5×5 linear filter in image processing, have a complexity factor of 40. More advanced algorithms usually have much higher complexity factors.

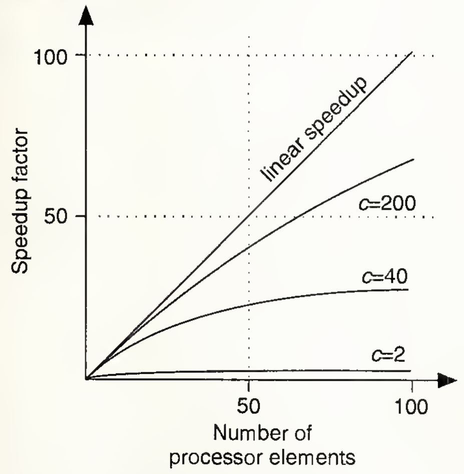


Figure A. Speedup factor as a function of the number of processor elements for different complexity factors c.

References

1. G.M. Amdahl, "Validity of the Single Processor Approach to Achieving Large Scale Computing Capabilities," *Proc. AFIPS Spring Computer Conf.*, Vol. 30, AFIPS Press, Reston, Va., 1967, pp. 483-485.

left neighbors (see Figure 4). Using the state of a clock counter and the current token, the processor element decides whether the present data value must be copied to the local memory. It likewise determines if the received value or a new value from its own local memory must be shifted to the right neighbor at the next communication clock cycle. A redesign of the logic cell array, currently under way, should increase communication speed to 15 MHz. The main objectives of the redesign are a better circuit layout of the LCAs and the use of pipeline techniques for processing tokens and data values.

The DSPs run at a 40-MHz clock rate with a peak performance of 60 Mflops. Notice that the DSP clock rate is independent of the communication clock rate. Up to 20 boards, or 60 processor elements, fit into a standard 19-inch rack,

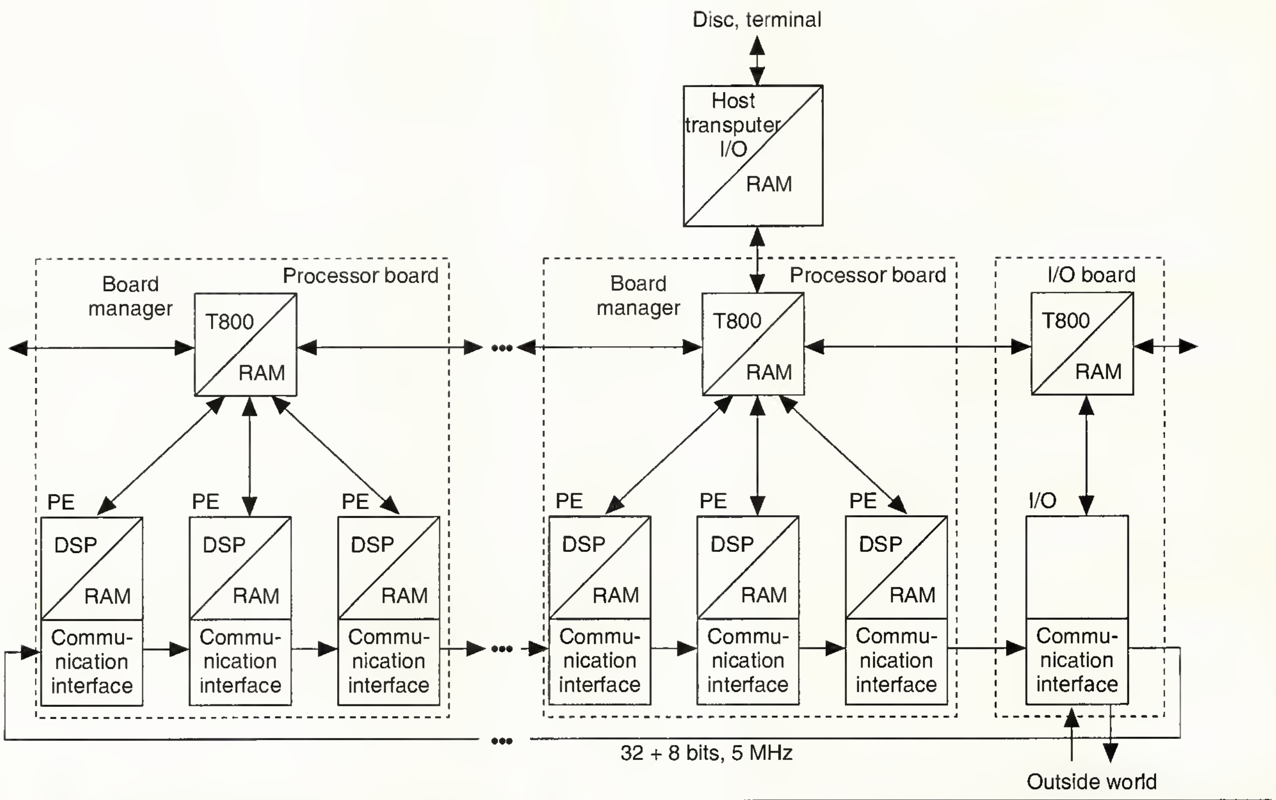


Figure 4. Overview of the Music hardware.

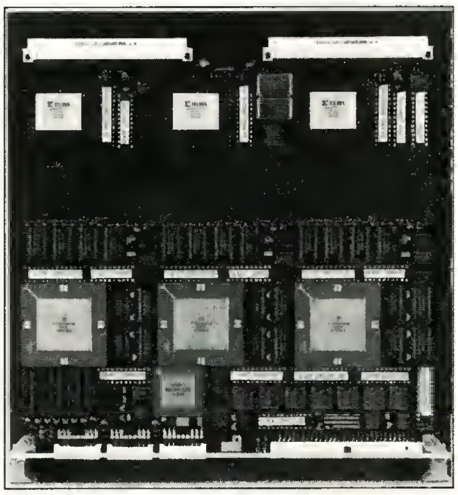


Figure 5. A Music board. The eight-layer board (double-height Eurocard: 8.6×9.2-inch) contains three processor elements. It has an arithmetical peak performance of 180 Mflops.

resulting in a 3.6-Gflops system with a hardware price of $22 per Mflop. Table 1 summarizes Music's technical data.

System programming

What do Music and its operating system look like from the programmer's point of view? A communication network cycle of data collection and data distribution is structured as follows: Data collection consists of copying all the subsets of the produced data from the various processor elements to one compact virtual-memory block circulating on the communication network. Data distribution then performs a new partitioning of this virtual-memory block and moves the resulting data subsets to the processor elements. The hardware fully supports data partitioning for one-, two-, and three-dimensional data sets.

To implement an algorithm on the Music system, we subdivide it into individual data-parallel iteration steps. *Data-parallel* means that all the processor elements carry out the same program but on different data subsets. Between two iteration steps, data redistribution takes place. Data partitioning is basically free, but the programmer usually tells the operating system along which axis (x, y, or z) to carry out the

Table 1. Music system technical data.

Number of boards	1	10	20
Processor elements	3	30	60
Peak performance (Mflops)	180	1,800	3,600
SRAM (Mbytes)	0.75 (3)	7.5 (30)	15 (60)
VRAM (Mbytes)	6 (24)	60 (240)	120 (480)
Power consumption (W)	35	350	700
Hardware price (US$)	4,000	40,000	80,000

Numbers in parentheses are maximum values. Present system size is 10 boards.

distributions. The operating system then determines the actual partitioning scheme at runtime. As a result, the compiled code is completely independent of the number of processor elements.

The operating system's standard partitioning method is to divide the data sets into equal pieces. This method works well if the processing time is independent of the data, which is the case in most neural network simulation applications. Another technique implemented in Music is dynamic load balancing, whereby the operating system distributes data according to each processor element's computation time in the previous iteration step. The present version of Music assumes that the computing-intensive areas of a data set differ little between two consecutive iteration steps. The implementation of other load-balancing paradigms in Music is conceivable.

Writing data-parallel code of an algorithm for a single DSP in the multiprocessor environment is almost the same as implementing an algorithm on a single-processor system. The only difference is that the DSP program must be able to produce only the part of the output data set specified by the operating system. A C compiler for the DSPs is available; time-critical parts of a program can be written in assembly language.

Implementing the algorithm so that it can work with parts of the data set and deciding along which axis to partition the data set are normally the only points where a programmer has to consider Music's parallelism. Special functions for downloading and uploading data between Music and the host computer automatically partition and distribute the data among the processor elements.

Back-propagation algorithm

One application of the Music system was developed for research on neural networks. (See the Mandelbrot set box for a short description of another application.) The first net-

The Mandelbrot set

The first program implemented on Music was the calculation of the well-known Mandelbrot set.[1] Named after Benoit B. Mandelbrot, a scientist at the IBM Thomas J. Watson Research Center, the Mandelbrot set lies in the center of the complex plane. To compute it, apply Equation B repeatedly to complex numbers.

$$\underline{z}_{n+1} = \underline{z}_0 + \underline{z}_n^2 \qquad \text{(B)}$$

Repeat Equation B until either $|\underline{z}_n| > 2$ or the number of iteration steps n exceeds a certain level (256 in the present implementation). The value of n then represents the gray level at the iteration's starting point $\underline{z}_0$ in the two-dimensional complex plane. Repeating this process for many different starting points produces beautiful pictures. However, calculating a complete image is computationally intensive. Table A compares Music and other systems in calculating the Mandelbrot set. All the implementations use exactly the same version of the algorithm, with 32-bit floating-point arithmetic, and the same part of the complex plane, with a resolution of 256×256 pixels.

References

1. H. Jurges, H-O Peitgen, and D. Saupe, "The Language of Fractals," *Scientific American*, Aug. 1990, pp. 40-47.

Table A. Comparison of processors in calculation of the Mandelbrot set.

Processor	Language	Time (s)	Performance (Mflops)
T800	Occam	22.0	1.5
80486	Pascal	20.0	1.7
TMS320C30	C	5.0	6.8
TMS320C30	Assembler	2.5	13.6
Music-1	Assembler	0.9	38.0
Music-10	Assembler	0.1	340.0

Music-1: a one-board (three-processor) system
Music-10: a 10-board (30-processor) system

The multilayer Perceptron and back-propagation

A widely used type of neural network is the multilayer Perceptron, the structure of which is shown in Figure B. The output, o_j, of a neuron is computed as

$$o_j = f\left(\sum_i w_{ji} \cdot o_i\right) \qquad \text{(C)}$$

where $f(.)$ is a nonlinear function (often the hyperbolic tangent); o_i denotes the inputs of the neuron, which are usually identical with the outputs of the neurons in the preceding layer (except the first layer); and w_{ji} denotes the weight that lies in the connection between the output of neuron i and an input of neuron j. Normally each neuron has an additional, constant-valued input, which serves as a bias or threshold input.

To train the network, we need training examples consisting of input patterns and the desired corresponding output vectors. First, we present an input pattern to the network. The input propagates through all the layers to the output of the network. The differences (errors) between the actual and the desired outputs are computed, and the weights are adapted so that the sum of the quadratic errors decreases the next time the same pattern is presented to the network. This procedure must be repeated many times with all the available examples.

A different version of the algorithm, called batch learning, updates the weights after the presentation of the complete training pattern set, not after each presentation of a training pattern. This technique minimizes the sum of quadratic errors over all training patterns in each training cycle. In many cases, however, the network converges significantly faster if the weights are updated immediately after each pattern presentation (continuous weight update). This is most often the case when training patterns contain redundancy; redundant information is used only once if the weights are updated at the end of all pattern presentations but can be applied many times in continuous weight update.

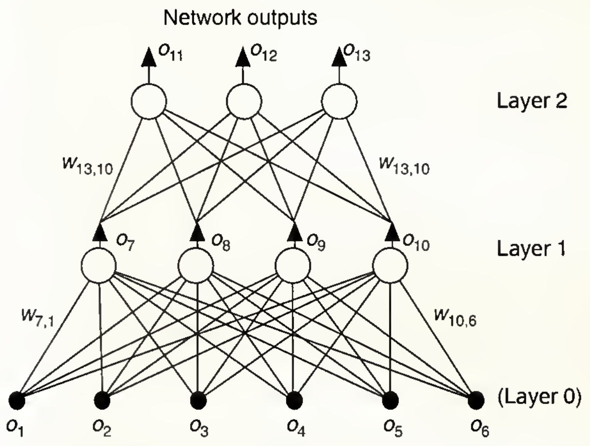


Figure B. Example of a multilayer Perceptron. The input layer is usually not counted since it performs no computation. Nevertheless, it is sometimes easier to view network inputs as outputs from the neurons of a zero layer. The neurons' outputs (o_i) have no index to denote the layer number because the layered structure is not explicitly required. That is, there could be direct connections between neurons of nonconsecutive layers. One restriction, however, is that network connections must not form feedback loops.

Equations D, E, and F give the rules for updating the weights ($w_{\text{new}} = w_{\text{old}} + \Delta w$) after the propagation of a training sample:

work structure we implemented is the multilayer Perceptron with back-propagation[6] (see Multilayer Perceptron box). Although there are many proposed modifications of the network structure and the learning algorithm to improve network performance, the standard back-propagation algorithm is still the fundamental method used by many researchers. It has been implemented on many computers, allowing a comparison of Music's performance with that of other systems.

Our implementation partitions the data as follows. In the forward path one network layer is computed after the other. The communication network first distributes a copy of the input pattern to all the processor elements. Each of them then computes a unique subset of the current layer's output vector. The communication network collects these subsets, and each processor element gets a copy of the complete output vector, which serves as input for the next layer. The processor elements save copies of the input vectors for later use in the weight-update process (Equation D in Multilayer Perceptron box).

In the backward path, each processor element first gets a copy of the complete Δ-vector (error measure) from the network's output layer. Each processor element then com-

The multilayer Perceptron and back-propagation (continued)

$$\Delta w_{ji} = \eta \cdot \delta_j \cdot o_i \qquad \textbf{(D)}$$

$$\text{with } \delta_j = f_j'(a_j) \cdot (t_j - o_j) \text{ for output neurons} \qquad \textbf{(E)}$$

$$\text{and } \delta_j = f_j'(a_j) \cdot \sum_k (w_{kj} \cdot \delta_k) \text{ otherwise.} \qquad \textbf{(F)}$$

The index k refers to all neurons connected to the output of neuron j, and t_j denotes the jth element of the desired output vector (also called the target vector) of the network. The constant η is the learning rate.

Note that Equation F, which computes the error measure Δ, is very similar to Equation C, which computes the output of the neurons. The algorithm is called back-propagation because this error measure is propagated backward through the network in the learning phase.

Normally the sum of the quadratic errors averaged over all training examples serves as a measure to control a network's learning progress:

$$e = \frac{1}{2n_p} \sum_p \sum_j \left(t_j^p - o_j^p\right)^2 \qquad \textbf{(G)}$$

where p is the training examples, n_p is the total number of training examples, j is the output neurons, t_j is the desired output of neuron j, and o_j is the actual measured output of that neuron. This measure is also the one that the back-propagation algorithm minimizes.[1]

References

1. D.E. Rumelhart, G.E. Hinton, and R.J. Williams, "Learning Internal Representation by Error Propagation," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, D.E. Rumelhart and J.L. McClelland, eds., Vol. I, Bradford Books, Cambridge, Mass., 1986, pp. 318-362.

putes a unique subset of the Δ-vector for the next lower layer. Afterward, the processor elements use the upper layer's Δ-vector once more to update that layer's weights. Figure 6 illustrates the partitioning scheme. Computation of the different vectors in this figure (output vectors in the forward path and Δ-vectors in the backward path) is split into equal parts carried out on different processor elements.

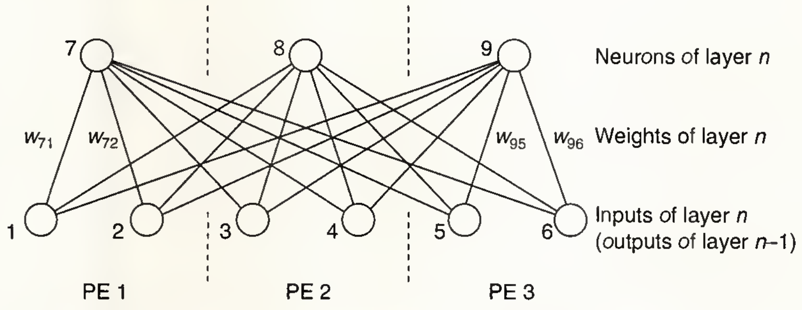
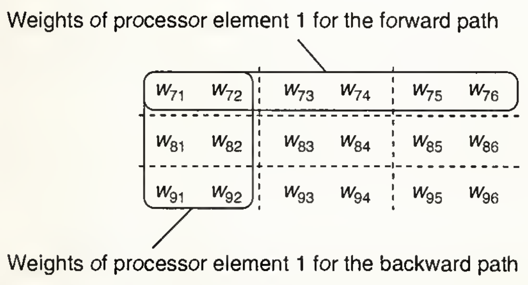


Figure 6. Implementation of back-propagation algorithm on Music.

Unfortunately, the implementation is not quite as easy as it first appears. The processor elements need different subsets of the weights to forward-propagate the input than to back-propagate the error measures Δ. Let's assume that a layer's weights are organized in a matrix $[w_{j,i}]$, where j denotes a neuron in the layer and i denotes the inputs of the neurons (Figure 7). This example shows partitioning for three processor elements. Different weight partitioning is needed for the forward and backward paths. In the forward path, each processor element must compute the outputs of a subset of the layer's neurons and therefore needs a horizontal slice of the weight matrix. In the backward path, each processor element must compute the error measured for a part of the neurons in the next lower layer and therefore needs a vertical slice of the same weight matrix.

Thus, each processor element must have local copies of two distinct subsets of the weights. The problem is that the



Weights of processor element 1 for the forward path
w_{71} w_{72} w_{73} w_{74} w_{75} w_{76}
w_{81} w_{82} w_{83} w_{84} w_{85} w_{86}
w_{91} w_{92} w_{93} w_{94} w_{95} w_{96}
Weights of processor element 1 for the backward path

Figure 7. Weight matrix corresponding to the layer in Figure 6.

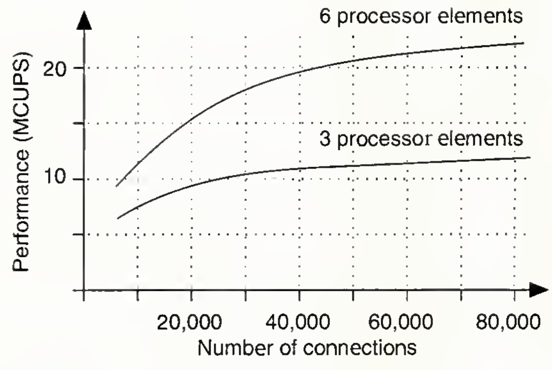


Figure 8. Performance of the back-propagation algorithm on Music for different network sizes (two-layer networks with equal proportions between the number of neurons in the layers).

weights can change after every learning step, so the network must collect and distribute the updated values after each back-propagation. This requirement dramatically increases the necessary communication because the number of weights rises almost with the square of the number of neurons in the network. Since the complexity factor of communicating the updated weights is very small (2 in the Music implementation), communicating the weights results in a very poor speedup (Equation A in earlier Performance box), with a maximum of 2. The system can avoid this communication saturation if each processor element updates both weight subsets individually.[7] Globally this means that every weight update will be carried out twice. However, it ensures that only the outputs of the neurons (while propagating) and the error measures (while back-propagating), but not the weights, must be communicated during learning.

The complexity factor of communicating only the back-propagated error measures is much higher than that of communicating the error measures and the updated weights. In the current Music implementation, for a layer of 100 neurons, for instance, it is greater than 200. The speedup then is close to linear for a wide range of the number of processor elements (see Figure A in the earlier Performance box). Notice that updating both weight subsets demands slightly more computing power than updating only one weight subset (a factor of 1.25 in the current implementation). This increase is not considered in Figure A, which illustrates the pure speedup. Nevertheless, the overall performance of a parallel system is significantly better if the weights are not communicated.

There are other methods for overcoming the weight-update problem in distributed memory architectures—for instance, the computation of partial sums.[8] The method just described, however, seems to be the most efficient one for the Music hardware.

For speed, we wrote the code for the DSPs entirely in assembly language. The DSP 96002 can perform a multiplication, an addition, a subtraction, and two data transfers (all floating-point) in one instruction cycle. The innermost loops of our implementation can use most of these parallel operations.

Results

In neural net simulations, performance is measured in millions of connections per second (MCPS) for the forward path and in millions of connection updates per second (MCUPS) for learning, including both the forward and the backward paths. Since each function needs a certain overhead to initialize its loops, performance increases for larger networks. Figure 8 shows measured performance as a function of network size for three- and six-processor Music systems.

Table 2 compares the performance of the Music implementation with results reported in the literature for the back-propagation algorithm and with our own performance measurements on different computers. As far as we know, all examples are based on floating-point computation. We determined the implementations' efficiency (the ratio of sustained algorithmic performance to the computer's peak performance, shown as a percentage in the table) by counting four floating-point operations per weight update in the first layer and six floating-point operations in the other layers. (In the first layer only the weight update, but no back-propagation, must be carried out.)

We emphasize that the Music system was designed for research and is therefore very flexible. That means it allows almost any modification of the network structure and learning algorithm. Other implementations are much more restricted. Many of the fast implementations[9-11] parallelize over the training set. Thus, each processor element has a local copy of the complete neural network and performs the back-propagation algorithm on a unique part of the training pattern set. Therefore, weight update is not possible after each presentation of a pattern but only after presentation of a large part of or the whole training set. In many cases, the effect of this strong restriction is that the learning converges much more slowly. Besides, it allows no freedom for research on the learning behavior of neural networks. Furthermore, holding a local copy of the complete neural network on each processor element limits the maximum possible network size, which does not increase with the number of processor elements as it does in the Music implementation.

Music's high computing performance is especially useful for neural network learning because the training phase is much more computationally intensive than the recall phase. That is because many training patterns (the number can easily reach several tens of thousands) must be presented to the network many times. In research one would also like to carry out numerous learning experiments while changing the network configuration each time. Figure 9 shows the result of such a series. The simulation on Music (with only six processor elements) took 56 minutes. The same simulation would take more than two days on a Sun 4 and more than a month on a Sun 3 workstation. A weekend of simulation time on a 30-processor Music system accordingly corresponds to about 1.5 years on a Sun 4.

Table 2. Comparison of back-propagation implementations.

System	No. of PEs	Performance MCPS	Performance MCUPS	Performance Peak (%)	Continuous weight update
PC (80486, 33 MHz)*	1	0.41	0.16	—	Yes
Sun 4*	1	1.1	0.51	48.0	Yes
Transputer T800[12]	64	27.0	9.9	—	—
RAP[8]	4	57.0	13.0	51.0	Yes
Warp[9]	10	—	17.0	—	No
CM-2[10]	64K	180.0	40.0	—	No
NEC SX-3**	1	—	130.0	9.6	Yes
Music-15*	45	427.0	203.0	34.0	Yes
GF11[11]	356	—	900.0	54.0	No

*Based on our own measurements. Music-10 denotes a 10-board (30-processor) system.

**Presented by N. Koike of NEC at the 1992 Second ETH-NEC Joint Workshop on Supercomputing (no published reference available).

MUSIC'S HARDWARE ARCHITECTURE allows the interconnection of a large number of processor elements, each with its own memory. The programmable communication network performs automatic distribution and collection of global data simultaneously with the activity of the processor elements. Able to process a great variety of algorithms, the Music system provides enormous computing power and yet retains the flexibility of a general-purpose computer.

Two real-world applications run successfully on Music. One is the back-propagation simulator we have described; the other is the simulation of molecule movement in a fluid. These applications confirm the system's efficient programming environment and speedup potential. An important reason for the success of the project, which is less than two years old, is the close cooperation of hardware development, operating system development, and application software development in our group of five persons.

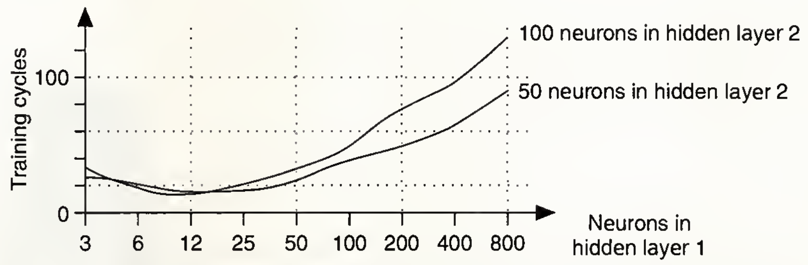


Figure 9. A three-layer neural network with 256 inputs and 10 outputs was trained many times with 600 handwritten digits. The number of neurons in the two hidden layers was changed each time. The diagram shows how many training cycles were needed to reach a mean-square output error of less than or equal to 0.03.

Our next step in neural network simulation on Music will be the implementation of a flexible function set and a corresponding command language, to give researchers a high degree of freedom in programming their own neural network structures and learning rules. The command language will have an interactive mode allowing experimentation with different network structures. We are also planning a compiler that translates a complete simulation job to DSP code, which then runs at full speed without the bottleneck of intensive communication between a command interpreter and the Music hardware. We will use this improved simulator for further neural network experiments.

Acknowledgments

We thank W. Van Gunsteren for promoting the Music project, H. Vonder Muehll for building and testing the hardware, W. Scott and A. Eichenberger for their valuable assistance in software development, and U.M. Franz, H. Walther, P. Guggenbach, P. Haag, M. Zehnder, and F. Buehlmann for their help in building and testing the Music system during their graduate work.

References

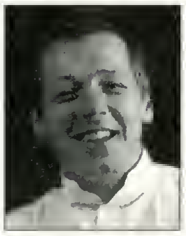
1. M. Annaratone et al., "The Warp Computer: Architecture, Implementation, and Performance," *IEEE Trans. Computers*, Vol. C-36, No. 12, Dec. 1987, pp. 1523-1538.
2. J. Beck, "The Ring Array Processor (RAP): Hardware," Tech. Report TR-90-048, International Computer Science Institute, Berkeley, Calif., 1990.
3. R.R. Shively and L.J. Wu, "Application and Packaging of the AT&T DSP3 Parallel Signal Processor," *Proc. Digital Signal Processing*, Elsevier, Amsterdam, 1991, pp. 208-213.
4. J. Beetem, M. Denneau, and D. Weingarten, "The GF11 Supercomputer," *Proc. 1985 Int'l Conf. Parallel Processing*, IEEE Computer Society Press, Los Alamitos, Calif., Aug. 1985, pp. 108-115.
5. "Connection Machine CM-2, Technical Summary," Tech. Report HA87-4, Thinking Machines, Cambridge, Mass., 1987.
6. D.E. Rumelhart, G.E. Hinton, and R.J. Williams, "Learning Internal Representation by Error Propagation," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, D.E. Rumelhart and J.L. McClelland, eds., Vol. I, Bradford Books, Cambridge, Mass., 1986, pp. 318-362.
7. H. Yoon, J.H. Nang, and S.R. Maeng, "Parallel Simulation of Multilayered Networks on Distributed-Memory Multiprocessors," *Microprocessing and Microprogramming*, Vol. 29, No. 3, Oct. 1990, pp. 185-195.
8. N. Morgan, "The RAP, A Ring Array Processor for Layered Network Calculations," *Proc. Int'l Conf. Application-Specific Array Processors*, CS Press, 1990, pp. 296-308.
9. D.A. Pomerleau et al., "Neural Network Simulation at Warp Speed: How We Got 17 Million Connections Per Second," *Proc. IEEE Int'l Conf. Neural Networks*, IEEE CS Press, 1988, pp. 143-150.
10. X. Zhang et al., "An Efficient Implementation of the Back-Propagation Algorithm on the Connection Machine CM-2," in *Advances in Neural Information Processing Systems (NIPS-89)*, D.S. Touretzky, ed., Morgan Kaufmann, San Mateo, Calif., 1990, pp. 801-809.
11. M. Witbrock and M. Zagha, "An Implementation of Back-Propagation Learning on GF11, a Large SIMD Parallel Computer," *Parallel Computing*, Vol. 14, No. 3, 1990, pp. 329-346.
12. H. Muehlbein and K. Wolf, "Neural Network Simulation on Parallel Computers," in *Parallel Computing 1989*, D.J. Evans, G.R. Joubert, and F.J. Peters, eds., North Holland, Amsterdam, 1990, pp. 365-374.

Urs A. Muller is a PhD student at the Swiss Federal Institute of Technology in Zurich, where he works in the Electronics Laboratory. Besides the simulation of artificial neural networks, his research interests include pattern recognition and real-time computer vision for robotics.

Muller received his MS degree in electrical engineering from the Swiss Federal Institute of Technology. He is a member of the IEEE.

Bernhard Baumle works as a research assistant at the Swiss Federal Institute of Technology, from which he received his MS degree in software engineering. His main interests include hardware and software architectures for parallel computer systems.

Peter Kohler also works as a research assistant at the Swiss Federal Institute of Technology. His main interests include hardware architectures and the design of parallel computers. He received his MS degree in electrical engineering from the Institute.



Anton Gunzinger teaches computer architecture and parallel processing at the Swiss Federal Institute of Technology. His research interests include real-time image processing and parallel processing.

Gunzinger received his diploma in electrical engineering and his PhD degree from the Swiss Federal Institute of Technology. For his SYDAMA synchronous dataflow computer, he received the Seymour Cray Prize. He is a member of the IEEE and the Computer Society.

READER SERVICE CARD

IEEE MICRO

October 1992 issue (card void after April 1993)

Name ______________________

Title ______________________

Company ______________________

Address ______________________

City ______________ State ______ Zip ______

Country ______________ Phone(______) ______

Reader Interest
(Add comments on the back)

Readers,
Indicate your interest in articles and departments **by circling the appropriate number** (shown on the last page of articles and departments):

150	151	152	180	181	182
153	154	155	183	184	185
156	157	158	186	187	188
159	160	161	189	190	191
162	163	164	192	193	194
165	166	167	195	196	197
168	169	170	198	199	200
171	172	173	201	202	203
174	175	176	204	205	206
177	178	179	207	208	209

Product Information
(Circle the numbers to receive product information)

1	21	41	61	81	101	121	141
2	22	42	62	82	102	122	142
3	23	43	63	83	103	123	143
4	24	44	64	84	104	124	144
5	25	45	65	85	105	125	145
6	26	46	66	86	106	126	146
7	27	47	67	87	107	127	147
8	28	48	68	88	108	128	148
9	29	49	69	89	109	129	149
10	30	50	70	90	110	130	
11	31	51	71	91	111	131	
12	32	52	72	92	112	132	
13	33	53	73	93	113	133	
14	34	54	74	94	114	134	
15	35	55	75	95	115	135	
16	36	56	76	96	116	136	
17	37	57	77	97	117	137	
18	38	58	78	98	118	138	
19	39	59	79	99	119	139	
20	40	60	80	100	120	140	

READER SERVICE CARD

IEEE MICRO

October 1992 issue (card void after April 1993)

Name ______________________

Title ______________________

Company ______________________

Address ______________________

City ______________ State ______ Zip ______

Country ______________ Phone(______) ______

Reader Interest
(Add comments on the back)

Readers,
Indicate your interest in articles and departments **by circling the appropriate number** (shown on the last page of articles and departments):

150	151	152	180	181	182
153	154	155	183	184	185
156	157	158	186	187	188
159	160	161	189	190	191
162	163	164	192	193	194
165	166	167	195	196	197
168	169	170	198	199	200
171	172	173	201	202	203
174	175	176	204	205	206
177	178	179	207	208	209

Product Information
(Circle the numbers to receive product information)

1	21	41	61	81	101	121	141
2	22	42	62	82	102	122	142
3	23	43	63	83	103	123	143
4	24	44	64	84	104	124	144
5	25	45	65	85	105	125	145
6	26	46	66	86	106	126	146
7	27	47	67	87	107	127	147
8	28	48	68	88	108	128	148
9	29	49	69	89	109	129	149
10	30	50	70	90	110	130	
11	31	51	71	91	111	131	
12	32	52	72	92	112	132	
13	33	53	73	93	113	133	
14	34	54	74	94	114	134	
15	35	55	75	95	115	135	
16	36	56	76	96	116	136	
17	37	57	77	97	117	137	
18	38	58	78	98	118	138	
19	39	59	79	99	119	139	
20	40	60	80	100	120	140	

SUBSCRIPTION CARD

SUBSCRIBE TO IEEE MICRO

All the facts about today's chips and systems

❑ **YES**, sign me up!

If you are a member of the Computer Society or any other IEEE society, pay the member rate of only $23 for a year's subscription (six issues).

Society: ______________________

IEEE membership no: ______________________

Society members: Subscriptions are annualized. For orders submitted March through August, pay half the full-year rate ($11.50) for three bimonthly issues.

❑ **YES**, sign me up!

If you are a member of ACM, ACS, BCS, IEE (UK), IEEE but not a member of an IEEE society), IECEJ, IPSJ, NSPE, SCS, or other professional society, pay the sister-society rate of only $39 for a year's subscription (six issues).

Organization: ______________ Membership no: ______________

For information on airmail option, send fax request to (714) 821-4010.

Full Signature ______________________ Date

Name ______________________

Street ______________________

City ______________________

State/Country ______________________ ZIP/Postal Code

❑ Payment enclosed *Residents of CA, DC, Canada, and Belgium add applicable sales tax.*

❑ Charge to ❑ Visa ❑ MasterCard ❑ American Express

Charge-card number

_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

Expiration date

_ _ _ _   _ _ _ _
Month   *Year*

Prices valid through 12/31/92
1092 MICRO

Charge orders also taken by phone:
(714) 821-8380 8 a.m. to 5 p.m. Pacific time
Circulation Dept.
10662 Los Vaqueros Circ., PO Box 3014
Los Alamitos, CA 90720-1264

Editorial comments

I liked: ______________________________

____

____

I disliked: ______________________________

____

____

I would like to see: ______________________________

____

____

Reviewers Needed. *If interested, send professional data to Dante Del Corso, Dipartimento di Elettronica, Politecnico di Torino, C.so Duca degli Abruzzi, 24, 10129, Torino, Italy.*

For reader service inquiries, see other side.

PLACE
POSTAGE
HERE

PO Box is for reader service cards only.

IEEE Micro
PO BOX 16508
NORTH HOLLYWOOD CA 91615-6508
USA

Editorial comments

I liked: ______________________________

____

____

I disliked: ______________________________

____

____

I would like to see: ______________________________

____

____

Reviewers Needed. *If interested, send professional data to Dante Del Corso, Dipartimento di Elettronica, Politecnico di Torino, C.so Duca degli Abruzzi, 24, 10129, Torino, Italy.*

For reader service inquiries, see other side.

PLACE
POSTAGE
HERE

PO Box is for reader service cards only.

IEEE Micro
PO BOX 16508
NORTH HOLLYWOOD CA 91615-6508
USA

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 38 LOS ALAMITOS, CA

POSTAGE WILL BE PAID BY ADDRESSEE

IEEE COMPUTER SOCIETY
PO BOX 3014
LOS ALAMITOS CA 90720-9804
USA

Walter Guggenbuhl is a professor of electronic circuit design at the Swiss Federal Institute of Technology and also serves as director of the Electronics Laboratory. His main interests are low-noise circuits and computer hardware for signal processing. Previously, he worked in electronic circuit and subsystem design as the manager of an R&D department at Contraves AG, Switzerland.

Guggenbuhl received an MS degree in electrical engineering from the Swiss Federal Institute of Technology and worked there as an assistant in the Department of Electrical Engineering while pursuing the PhD degree. He is a member of the IEEE and the Computer Society.

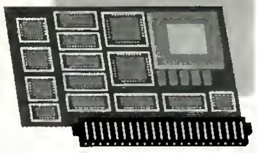
Address correspondence about this article to the authors at Electronics Laboratory, Swiss Federal Institute of Technology, ETH Zentrum, CH-8092 Zurich, Switzerland; e-mail: urs@st.ife.ethz.ch.

Reader Interest Survey

Indicate your interest in this article by circling the appropriate number on the Reader Service Card.

Low 168 Medium 169 High 170

On the Edge



Tools to make the engineer's life easier

[Carl Warren coauthored this column shortly before his death last July. It continues a series of columns describing personal experiences with devices, software, or ways to solve problems. Coauthor James Gafford introduced the series in June 1991 with a discussion of rate monotonic scheduling.]

Carl Warren and James Gafford
McDonnell Douglas Space Systems Company

Carl Warren

McDonnell Douglas

Space Systems Company

(714) 896-3311

x. 7-1230

warren@ssdunx.mdc.com

Whether you're a hardware or software engineer, you are no doubt faced with information overload. Some of the information we (engineers) deal with comes from interesting articles in trade magazines (that you surely would like to reference later), technical brochures, and even your own meanderings in the lab or with pencil and paper.

No matter how good your intentions, or even if you pay due attention to notebooks and what not, information gets lost in the shuffle.

While preparing our plan for this series of columns, we "guesstimated" that between us we had probably filled more than 500 notebooks, dutifully labeled thousands of file folders, and even went to the trouble of creating paper and computer indices of our valuable data. Since we are systems engineers and computer scientists, we naturally feel we have a lock on the proper way out of this dilemma—hence this series of columns.

We hope you find this series of columns, on making use of automation and software tools, interesting and informative. Our goal is to make your life easier.

This column isn't what you would suspect (a clever software solution); rather it deals with your library. Since many readers of *IEEE Micro* tend toward the hardware side, there is a need to create a small foundation for discussion. Therefore, we decided to suggest various books and reference materials you might want to consider for your library. Notice we don't quote prices or suggest booksellers. The books we talk about range from $45 to $100 plus, depending on where and how you buy them. You might find your company librarian could purchase these books for your use at a significant discount, and once in the library you and your teammates have equal access.

Even though most of us worried our way through hours and hours of physics courses and a pile of mathematics to become engineers, these disciplines easily atrophy if not exercised on a regular basis. Unfortunately, modern engineering doesn't necessarily cause us to rigorously exercise our mathematical or physical brains. An interesting, if not fun, solution is to revisit some of what we learned in our undergraduate days. Although most of us plowed our way through Resnick and Halliday—a respected physics tome that has stood the test of time for more than 25 years—a new physics textbook series is available from Saunders Golden Sunburst Press, Philadelphia. This multivolume set, *Physics for Scientists and Engineers*, written by Raymond A. Serway, is designed as a modern university-level teaching tool. (The basic set for a three-semester course is two volumes, with a third for advanced courses.) The multicolor books are supported by well-thought-out viewgraphs, software examples on diskette, instructor manuals, and a thoughtful study guide. The books are chockfull of useful tables, equations, excellent

0272-1732/92/1000-0066$03.00 © 1992 IEEE

examples of Faraday's laws, and interaction with the "real" world.

The Serway physics books hold a prominent place in our library as a ready source of information on physical phenomena. We use the books in much the same manner as the *Radio and Electronics Engineer's Handbook*, but we can double-check to see if we are handling the mathematics correctly.

Don't forget the math books

Admittedly, most of our work concerns hardware/software integration and testing. As a result we do very little basic hardware design, so we need only a few books in that area. However, we do need to keep abreast of various books and tools that help us in designing software models and creating simulations.

Possibly, one of the landmark books for people who muck about with numbers and computers is *Numerical Recipes in C,* by William H. Press, Harvard-Smithsonian Center for Astrophysics; Brian P. Flannery, Exxon Research and Engineering Co.; Saul A. Taukolsky, Department of Physics, Cornell University; and William T. Vetterling, Polaroid Corp. (Cambridge University Press, New York; copyright 1988). This book was originally written for Fortran and Pascal; but according to the authors the demand for a C version ranged from polite to demanding. They are still considering Basic and possibly a C++ version.

You can probably guess just from the author's affiliations that this is not a trivial book. Indeed it provides you with C routines (all available on a diskette that accompanies the book) for everything from the Gauss-Jordan elimination to multidimensional integrals, eigensystems, FFTs, and my personal favorite, diffusive initial value problems.

Besides the depth of information contained in the book, from a purely mathematical standpoint, the authors don't leave you hanging. Even if you're not "mathematically mature," you can use the routines and examples given. For example, since we design specialized simulations for testing, we frequently need to take snapshots of data distribution and relate it in some manner. One method is Kendall's Tau that allows us to view information based on relative order of ranks to tell us where the "weaker" properties of the data are. The distribution formula is

$$\mathrm{Var}\ (t) = 4N + 10/9N(N-1)$$

where N is the number of data points, and t is the variable distribution within the ranking margins.

This can be a pretty grisly piece of computing if handled in a linear fashion. *Numerical Recipes* gives you a routine that plops the various distributions into bins, thus reducing the processing time—something that you probably wouldn't have the intuitive mathematical or programming knowledge to do.

Another book that you should give serious thought to is *Microcomputer-Based Numerical Methods for Science and Engineering* by Gary J. Lastman, University of Waterloo, Ontario; and Naresh K. Sinah, McMaster University, Hamilton, Ontario (Saunders College Publishing, New York; copyright 1988). This book, designed as an advanced university-level course, is available with a diskette of all problems and solutions in Pascal and an instructor's solutions manual.

We recommend this book for those of you who are just now reeducating yourselves on mathematical methods and learning scientific and engineering programming. While *Numerical Methods* is a great source book for the professional, this book provides the novice with a solid foundation.

The authors spend time in the early chapters describing errors in input and output data, rounding off, and ways to manage the mathematical model. They also discuss floating-point math and the way it is represented for various applications. This book should be treated as a learning source from which you will become familiar with handling nonlinear equations, curve fitting, ordinary differential equations, and minmax of functions.

With so much to absorb in this book, you may find it worthwhile to spend an hour or two in the evening spitshining your "mathelogical" skills before tackling the next tasks on the computer agenda.

Of course, we would like to recommend other books and resources in the area, but we have a finite amount of space.

Fundamentals of computer science

In this area we can provide you with more information than you would probably ever want—or for that matter need. So we have chosen a few books that you should consider reading and studying to acquire a basic background.

Unfortunately, people assume—and engineers tend to do this more often than not—that to create a database or a data logging system, you simply cobble together the routines and "ride for the Red River." No doubt sometimes simple and sweet will work, but most likely you will need to concern yourself with how the data is structured. Sometimes you need to consider linked lists, queues, matrices, and how to get the best performance from your system.

In this case *Data Structures with Ada* by Michael B. Feldman, George Washington University (Reston Publishing Co., Reston, Va.; copyright 1985) is just the book you need to read. This book is considered the landmark book for most computer science courses. The use of the Ada language is significant since it is the US government's choice of language for large projects, and it does serve as a convenient tool-building guide to understand how data structures work.

Similarly, *File Structures with Ada* by Nancy E. Miller and Charles G. Petersen, Mississippi State University

What's #1 in the hearts of its readers?

IEEE Micro!!

How does it do it?

With quality articles!

Who keeps the quality high?

The article reviewers!

Become a reviewer and join the #1 team.

Want to help keep *IEEE Micro* #1? Editor-in-Chief Dante Del Corso seeks more technical reviewers—people interested in seeing that the articles published in *IEEE Micro* continue to be of the highest quality. The technical review process is a crucial step in providing you with correct and timely information so you can keep up with your ever-changing profession.

Send your professional information directly to:
Editor-in-Chief Dante Del Corso
IEEE Micro
Dipartimento de Elettronica
Politecnico di Torino
C.so Duca degli Abruzzi, 24
10129 Torino, Italy
delcorso@polito.it

(Benjamin/Cummings Publishing Co., Inc., Redwood City, Calif.; copyright 1990) helps you understand file structures, records, access techniques, file manipulation, and management of storage devices. You also learn how to organize data and perform sorts and merges—all functions needed to create an information system.

The examples and exercises in both books are excellent and well worth pursuing. If you're planning an in-house training program for Ada, and system design with Ada, you probably will want to use these books as the course foundation.

One book that every computer scientist or serious designer has at the workstation is *Computer Algorithms: Introduction to Design and Analysis, Second Edition,* by Sara Baase, San Diego University (Addison-Wesley Publishing Co., Reading, Mass.; copyright 1988). The key to getting the most out of your system is how well you implement the rules of doing work. This book describes what an algorithm is, how to analyze a problem and describe it in terms of a solution "algorithm," and how to implement it.

Baase's book treats everything from mathematical analysis to sorting to creating parallel algorithms. We do need to warn you, however, that this is not an easy book to understand and does require a quiet room and lots of study. The book is mathematically rigorous in the treatment of the subject. Although Baase is thorough in her treatment, the prose is terse and consequently becomes a little mind-boggling at times. These aren't shortcomings—it just means you have to spill extra brain juice.

Naturally there are other books, and for that matter, articles that we would like to treat you to. But our space is just about gone.

What's coming

Now that you have some idea of what books you might want to buy, you need something to keep all this information in. The next several columns hold some surprises for you and some useful tools. For example, James Gafford, the resident McDonnell Douglas maven of Hypercard style tools, will be showing you how to get the most out of Hypercard for the Macintosh and Asymmetirx Toolbook for Windows and Oracle card for both environments. He's even gone to the trouble of automating the *Ada Handbook* for you.

Additionally, we will be showing you how to make use of your favorite database manager, such as Borland's Paradox, and Fox's (now possibly by the time you read this Microsoft's) Fox Base for creating "information" buckets. In concert with making use of database managers, we show you how to use Borland's powerful Object Vision to produce useful applications for managing your information bases. And, yes, we have created some interesting databases that you can use right now.

James D. Gafford, a senior engineer/scientist at McDonnell Douglas Space Systems Company, graduated from the US Naval Academy with a degree in aerospace engineering. He has worked for 13 years in the spaceship industry.

Reader Interest Survey

Indicate your interest in this department by circling the appropriate number on the Reader Service Card.

Low 192 Medium 193 High 194

continued from p. 8

military technology supported by the Japanese navy and army, while they were not supported by widespread civil technologies. The reason why they were not supported by civil technology is simple. They could not be supported, because there was not widespread civil technology in Japan, which includes mass production technology.

In the short term, military technology can show dazzling progress when military technologists utilize the ample fruits of civil technologies. They can spend limitless money and human resources. This is because military technologies are vital for a nation's survival. With such conditions and environment, it is natural for military technology to show rapid progress.

We should, however, take note that military technologies are those necessary for limited areas, and as such are a showcase of the best technologies.

After World War II the Japanese abandoned the maintenance and fostering of aggressive military force and technologies. And the country decided to live by peaceful economic development. In contrast to prewar Japan, expenditure for military technology development in Japan is now negligible.

Present technologies are basically ones for civil use, and these have rapidly progressed. When the Gulf war broke out, the US army demonstrated overwhelming strength over the Iraqi army, giving a strong impression that US war machines were equipped with highly advanced technologies, which permitted them to easily defeat the Iraqi military force.

With the overwhelming superiority of weapons (which had been supplied by the USSR, China, and European countries), another fact struck the Japanese. It was reported that the US military force could not sustain its power without Japanese technology; that is, the US military force could not prepare weapons required for the front without Japanese assistance. No clear evidence was demonstrated, so it is impossible to define to what extent Japanese technologies were applied in those weapons used in the Gulf war, whether the report was true or not, or whether such claims are exaggerated.

Anyway, there might be some facts by which such a rumor could be generated. These facts suggested that civil use technologies are sometimes at a very high level equivalent to the most advanced military technologies.

Consumer market forces are ultimately more successful than military in the development of state-of-the-art technology.

Home appliance technology. Much of the high technology found in Japanese homes is that which forms the basis of Japanese industrial technologies. For instance, the gallium arsenide element was first proposed for military and satellite use. When the Japanese tried to apply the GaAs element for the home VTR, demand for the GaAs element jumped. In a few years, demand for GaAs elements increased from 10,000 to 1 million pieces per month, while demand for GaAs elements in each European country remained at less than 10,000 per month.

As demand for GaAs elements increased rapidly, Japanese manufacturers were confronted with various problems such as improvement of yield, preparation of high-quality wafers, processing technology development, development of high-performance ICs, and so on. Thanks to the large demand, they could spend a lot of money on R&D. According to the common sense of the US and European technologies, home-use VTRs did not warrant such advanced technology as represented by the GaAs element. A VTR was merely a home-use electric appliance, while GaAs elements were being developed for ICBMs, satellites, etc., which require very high performance regardless of element cost.

Japanese industrialists disregarded such reasoning and began to manufacture low-performance GaAs elements. They continued R&D until at least they succeeded in developing satisfactory GaAs elements. Development cost per unit of the GaAs element in Japan was very small; however, total expenditure was large. This permitted Japan to become the world's leading GaAs manufacturer. It is reported that many countries advanced in military technologies are now using Japanese GaAs elements in their weapons.

Japanese automation and robot technologies have been developed by several groups, one of which is that led by watch manufacturers. This group developed a high-precision automation system that could assemble watch mechanisms. Assembly-line technologies of electronic appliances were developed mainly based on the technology. The first Japanese markets for carbon fibers and advanced ceramics were fishing rods, tennis rackets, home-use health care magnets, and necklaces. Japanese manufacturers developed markets for their high-tech products in civil or home-use products. In sustaining the market necessary for continued manufacturing and R&D, home use product markets are indispensable.

The fact that many home appliances are incorporating high technologies is, in other words, because home appliances are changing continuously. The audio hardware and software market is a good example. Records have almost disappeared from records shops, and production of turntables has decreased dramatically, while that of CD players has jumped fantastically. Major electric home appliances is another example. Many electric home appliances have had dramatic production increases in the last 10 years. These include high-fidelity VTR, heat pump air-conditioners, large-screen TV sets, fully automatic washing machines with dryers, CD players, camcorders, electronic ranges with oven, and large refrigerators.

There are other examples in which standard products have had new technologies applied to them. However, the major performance required for these new products has not changed; for instance, a fully automatic washing machine can only wash clothing the same as the old-fashioned washing machines. These products have changed corresponding to changes in people's needs.

The electronic cooling machine seems to have no relation with high-technologies; however, it is in fact a product crammed with high technologies. These include 1) fuzzy-control microprocessors and software that permit automatic cooking of some 400 dishes; 2) one-chip LSIs that integrate all peripheral circuits necessary for the cooking machine; 3) very light, high-voltage 4-kV transformers; 4) 30-kHz inverters that control output; 5) high-power 1-GBT element; and 6) high-performance noncontact sensors that can measure temperature correctly in the environment when a 2-GHz wave is being emitted.

These sensors include pyroelectric IR sensors that are being applied for homing devices of missiles and gas temperature sensors that measure temperature of gas emitted from foods as well as other vapors. In addition to these, a cooking machine now employs a mechanical weight sensor that measures the weight of foods, a smell removal unit that prevents the smell of food in rooms, and high-performance paint that permits the sweeping away of contamination with one wipe of a surface.

The camcorder is also a typical product in which a lot of high-tech devices are being used. Miniaturization of camcorders has permitted use by many people so that the market has greatly expanded. A camcorder contains a camera and VTR with battery and plenty of functions in a compact case. To realize such miniaturization, high-density recording, high-density mounting, multipixel image elements, and micromechanism technologies were applied.

Manufacturing of high-tech products at low cost requires very advanced technologies.

HDTV has been commercialized for a limited area, though wide use for broadcasting has not yet begun operation. However, it will be commercialized this year in Japan. In HDTV, a variety of high technology is being applied, including advanced LSI technology, satellite broadcasting, high-definition imaging, and image processing. In comparison with the new products just listed, many home electric appliances are decreasing in product number even though new technologies are incorporated. These include open-reel tape (replaced by cassette recorder), needle record player (replaced by CD player), and monochrome TV (replaced by color TV).

Why home appliances promote technologies. Military technology demands utmost efforts by a nation, both in money and human resources, to promote progress of technology to defend lives on its side and to win in war. Then what is the driving force for the development of home appliance technology? Apparently, it is the necessity for survival of companies in business competition. The following factors can be listed as some of those supporting progress in technology.

Large-volume manufacturing. ICBMs, satellites, space shuttles, and missiles require very, very high technologies. However, the number of units to be manufactured is limited. As mentioned earlier, electronic cookers employ pyroelectric IR sensors that are being used for homing devices of missiles. There is no difference in the two parts. Some 1.5 million electronic cookers are being manufactured every year, yet nowhere near as many missiles are manufactured in any country.

R&D on sensors is being carried out every year. Is it possible for a missile developer to compete with Japanese cooker manufacturers in the field of pyroelectric IR sensors? The CCD (charged coupled device) element used in camcorders is basically the same as that used for small missiles and ICBMs. Meanwhile, some 5 million CCD elements are being manufactured for camcorders. The fuzzy control system is widely used for consumer products, though its function is limited and shows low performance. The number of units in which the fuzzy control system is used totals over 10 million. Even when investment in R&D for consumer products per unit is small, total investment is very large, which permits rapid progress in technology.

Low-cost manufacturing. The price of parts used for consumer products must be low. Success in manufacturing products that satisfy all requirements does not mean that the R&D is successful. Technologists must develop a manufacturing process and design that permit the manufacture of such products with satisfactory performance at a cost consumers can afford. Manufacturing of high-tech products at low cost requires very advanced technologies indeed.

Developing a product accompanied by a manufacturing system. Completion of development work of consumer products implies that implementation of a volume production line has been accomplished. The consumer product is not completely designed until the volume production facilities are completely designed and developed. There are many examples of useful products that failed to debut due to failure in establishing volume production technology.

Users lacking technological knowledge must be able to operate new products. As electric appliances and other consumer products are used by just about everyone, the products must be easy to use. These are not products that only skilled specialists operate, such as weapon systems for trained soldiers.

A commodity must demonstrate full-scale performance even when operated by a child. As a result, technologies being applied to such a commodity must be extremely refined. This fact requires machine designers and commodity planners to employ the latest technology.

[The report then gives several very lengthy examples of the development of consumer goods that apply high technologies, including fuzzy control for home appliances, high-density surface mount technology, home elevators, paper diapers, noise reduction in appliances, and lighting.—D.K.K.]

Reader Interest Survey

Indicate your interest in this department by circling the appropriate number on the Reader Service Card.

Low 195 Medium 196 High 197

Micro Review

Richard Mateosian

2919 Forest Avenue

Berkeley, CA 94705-1310

(510) 540-7745

Look it up!

The publication of the Third Edition of the *American Heritage Dictionary* inspired me this month. I've looked at that dictionary and an on-line dictionary as well.

American Heritage Dictionary of the English Language, Third Edition (Houghton Mifflin, Boston, 1992, 2,184 pp.; $39.95)

I have been using the *American Heritage Dictionary* for a long time. I have always liked it, but I have a few reservations about it too. This new edition improves greatly on the older editions. I have some of the same reservations, but they aren't big ones. I like to use this dictionary to look up words, and every time I do, I spend a little time browsing through other entries. This dictionary is fun to read.

Anne H. Soukhanov, the executive editor, led a team of 175 contributors in a four-year project. They used a database of hundreds of millions of lines of text. They referred thorny questions of usage to their usage panel, a group of 173 writers, editors, and scholars, chaired by Geoffrey Nunberg of Stanford University. The panel, which included people like Isaac Asimov, Letitia Baldridge, Julian Bond, William F. Buckley, Jr., Alistair Cooke, Eugene McCarthy, Carl Sagan, Glenn Seaborg, Wallace Stegner, and Eudora Welty, voted and commented on these issues. This feedback formed the basis of the many usage notes throughout the dictionary.

The editors also sought counsel from more than 60 specialists in areas as diverse as dialect, sports, music, religion, science and technology, North American Indian languages, and trademarks. The only identifiable computer scientist among these specialists is Donald C.S. Allison of Virginia Polytechnic Institute and State University.

In her introduction, Soukhanov identifies the achievements that she feels made the first edition of the dictionary notable:

> It faithfully recorded the language in easily understood definitions. It provided guidance toward accuracy, precision and grace in the use of English that intelligent people need and seek in a dictionary. It traced, whenever possible, English words to their origins and keyed many to an Appendix of Indo-European Roots. And it presented complex lexical data in a typographically attractive design accented by thousands of photographs and line drawings in spacious margins.

The new dictionary maintains the look and feel of the earlier editions.

Each two-page spread has four columns, two and seven-eighths inches wide, and two one-and-three-quarter-inch outer margins. Each column contains approximately 25 boldfaced entries and their senses (definitions). The margins contain illustrative pictures, an average of around four per spread. In addition, one of the margins of each spread contains a summary of the pronunciation key, so you can learn the pronunciation of any entry without turning the page.

The editors have compiled a dictionary that focuses on the current state of English. This contrasts with the historical perspective of some other dictionaries. They used their database of citations to help them determine current usage. The definitions for each entry begin with the "most central" sense of the term. Others follow in a "rational order."

I really appreciate the usage notes in this addition. They deal with issues as varied as syntactic subtlety and political correctness. For example, the usage note at the entry for *majority* explains when to use a singular verb (The majority elects the candidate it wants.) and when to use a plural verb (The majority of voters live in cities.). The usage note at the entry for *black* discusses the pros and cons of capitalizing it in sentences like "Together, blacks and whites can move beyond racism."

Some usage notes present the opinions of the usage panel. For example, the usage note about *world-class* reports that 65 percent of the panel accepted the usage "world-class restaurant" but only 4 percent agreed with the usage "AIDS is a world-class tragedy." When I first started to use this dictionary, this kind of voting offended me. I was annoyed that 30 percent of the panel of the original edition would use *loan* as a verb. Nowadays I don't feel that way. The language is continually changing, and I value the opinions of other thoughtful people about what is happening at the fringes.

In addition to usage notes, the dictionary accompanies many entries with word histories. For example, the word history note at *nerd* considers whether or not the term originated in Dr. Seuss's *If I Ran the Zoo*. An illustration of a Nerd from that book appears in the margin.

One of the ways Soukhanov feels the *American Heritage Dictionary* distinguishes itself is its use of computer technology. Although the dictionary begins with several interesting articles about the English language and the dictionary, it unfortunately does not contain an article about their "complex, highly versatile, structured database." This database contained hundreds of millions of lines of text, drawn from sources as varied as Shakespeare and *The New Republic*.

The dictionary doesn't do a very good job with computer terminology. It contains a helpful usage note at *data*, but it has nothing to say about the use of *media* in the context of data storage. Its entries for *MS-DOS* and *UNIX* contain a strange distinction. One is "a trademark for a microcomputer operating system," and the other is "a trademark used for a computer disk operating system." I'll let you try to guess which is which.

The dictionary's definition of *structured programming* is "a method of designing and writing programs in which the statements are organized in a specific manner to minimalize error or misinterpretation." This definition misses the whole idea of stepwise refinement of the algorithm and its associated data. It focuses instead on the fomat of the statements. Perhaps wisely, the dictionary contains no entry at all for *object-oriented programming*.

The dictionary contains many names of famous people, so I looked up a few from the computer field. John von Neumann is identified only as a Hungarian-born mathematician noted for his contributions to game theory and quantum theory. Anyone wondering what a von Neumann machine is will have to look elsewhere. Alan Turing doesn't appear at all. Neither do Backus, Dijkstra, Kay, Noyce, and a variety of others. The only adequate entry I found was for Shockley, the inventor of the transistor.

I went through a few pages of the Microsoft Press *Computer Dictionary* (see Micro Review, August 1991, pp. 42-45) and looked up each word in the *American Heritage Dictionary*. The Microsoft dictionary was more comprehensive and usually had better definitions.

To summarize, the *American Heritage Dictionary* is a wonderful piece of work—comprehensive, informative, pleasing to the eye, fun to read—but if you want to look up computer terms, look elsewhere.

Instant Definitions (Word Science Corp, 1415 Oakland Blvd., Suite 220, Walnut Creek CA 94596; (510) 939-1190; $69.00)

This product, which was originally called Definitions Plus, is an on-line version of the *American Heritage Dictionary Office Edition*. It contains brief definitions of 116,000 words. You invoke the program once, and then 4 Kbytes remain resident until you reboot your computer. Three hot keys, which you can assign, are supposed to make Instant Definitions functions available from within any text-based program, such as Word Perfect. I was unable to use it from Brief, but perhaps I could have done so by experimenting with different hot keys. Unfortunately, by the company's own admission, Instant Definitions doesn't work well with Windows or with other terminate-and-stay-resident (TSR) programs. On my system I had to reboot every few times I used it.

The program is fun to play with and potentially extremely useful. All you have to do is position the cursor near a word or phrase you're interested in and press the appropriate hot key. A small window (in a color scheme you can select) appears, giving the definition. You can scroll through definitions that don't fit in one window. If the word you have selected is not in the dictionary, a window of suggestions appears. You can step through these, and a separate small window shows a summary definition of each as you do so. If you reach one you like, you simply hit Enter, and a definition window opens for it. Everything is truly instantaneous. There are no perceptible delays.

Instant Definitions has several other nice features. You can select a word within a definition window and ask for its definition. Another definition window appears elsewhere on the screen. You can continue this indefinitely, and a history feature remembers the last 50 or so entries. You can step back through them using a mechanism similar to the one used for suggestions.

Instant Definitions uses the same stepping mechanism for two other interesting features. You can start at the current word and browse alphabetically through the dictionary, or you can create a list to browse through by telling Instant Definitions to search for entries containing specified words. For example, if you tell it to search for *red flower*, it will form a list of all entries containing both the word *red* and the word *flower* in the definition.

I spoke with Matthew Dalton, the president of Word Science, and he told me about their two-pack donation program. Every buyer of the program receives two legal copies of the software (but only one copy of the documentation). You can give your extra copy to anyone you like, but Word Science encourages you to give one to a school or literacy center. If you wish to do this in an organized way, you can donate your extra copy to Gifts in Kind America of Alexandria, Virginia. Their telephone number is (703) 836-2121. I hope other software publishers will adopt similar programs.

I think this is a wonderful application of computer technology to document generation, but if you decide to buy it, you may have some work to do to make it compatible with your system's TSRs and word processors.

Reader Interest Survey

Indicate your interest in this department by circling the appropriate number on the Reader Service Card.

Low 183 Medium 184 High 185

New Products

Send announcements of new microcomputer and microprocessor products to Managing Editor, IEEE Micro, *PO Box 3014, Los Alamitos, CA 90720-1264.*

Joe Hootman

University of North Dakota

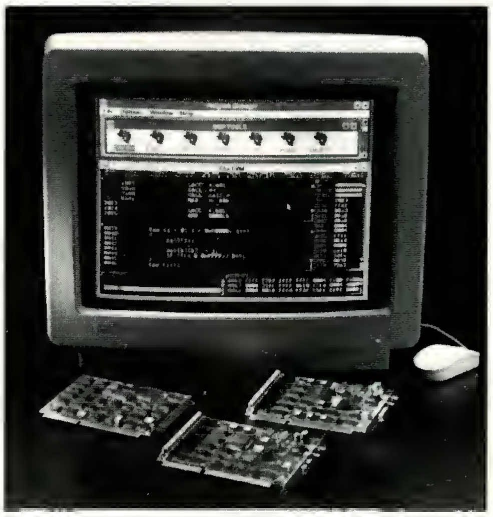
DSP software, hardware

TMS320 development tools

Three development tools help users of digital signal processors design and test applications and deliver them to market quickly. The evaluation modules support device evaluation, benchmarking, and system debugging for 16-bit TMS320C5x, TMS320C2x, and TMS320C16 processors.

The PC/AT plug-in cards include the appropriate DSP; 64 Kwords of fast SRAM; and the high-level-language, mouse-driver TMS320 Programmer's Interface, which eliminates the need to memorize complex commands. Also included are assembler/linker packages, an expansion port for interfacing to peripherals, and a backplane serial port for communication to other hosts, systems, and target boards. Users can define windows for visibility into the specifics of the device. *Texas Instruments; from $795.*

Reader Service No. 11



DSP evaluation modules from Texas Instruments

Convert signals up to 200 kHz/channel

Two 16-bit, dual-channel daughter modules allow designers of DSP-based systems to convert analog signals up to maximum sampling frequencies of 200 kHz per channel for a broad range of buses. When attached to a motherboard, the modules do not violate the single-slot requirement for each of the bus standards. The modules may be used alone or to provide analog I/O capability for the company's PC, VME, or Sbus DSP motherboards that are equipped with module sites.

The AM/D16SA successive approximation module contains on-board antialiasing and reconstruction filters that can reconstruct digital samples at up to 500 kHz per channel at SNR 90 and 86 db. The second module, AM/D16DS, contains Delta-Sigma A/D and D/A converters that operate at a maximum sampling frequency of 50 kHz. It's input and output signal-to-noise ratios are 90 dB. *Spectrum Signal Processing; $995 each, both versions.*

Reader Service No. 12

Simultaneously sample four channels

MAD100 provides IBM PCs and compatibles with simultaneous data sampling of up to four channels. In single-channel mode, the card samples at 100 MHz; additional modes sample at 50 and 25 MHz. Users can program the trigger source as internal (one through four channels), external analog, or external TTL; the trigger offset is also programmable. MAD100 features programmable input gain from 640 mV to 64V full scale, memory depth to 256K, and pretrigger data capability up to the full memory depth. *Markenrich Corporation; $2,995 each.*

Reader Service No. 13

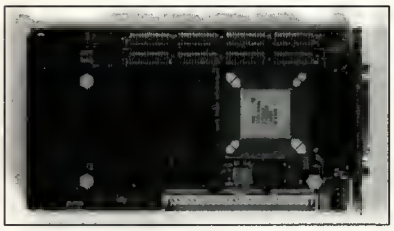
Sbus board contains SCSI controller

An Sbus board based on the 33-MHz TMS320C20 floating-point DSP supports applica-

0272-1732/92/1000-0074$03.00 © 1992 IEEE

tions that need to read and write data without consuming all of the Sun disk resources. Designed with an NCR53C94 advanced SCSI controller, the 2-Mbyte/s I/O board allows interprocessor communication between Sbus C30 boards. It provides 512 Kwords of SRAM, dual-port RAM, and analog-to-digital options. *Spectrum Signal Processing; $4,795 each (SCSI board), $9,795 each (Sun 4 compiler, monitor, and Sun OS drivers), $995 each (analog I/O options).*

Reader Service No. 14


Spectrum Signal Processing's Sbus board

PC chip sets feature sound

DSP-based audio chip set family features the company's music synthesis technology, which is based on samples of actual musical instruments stored in the chip set. The three PC products range in sound quality from emulation of the Sound Blaster music board through the Aria Synthesizer that uses a 512-Kbyte or 1-Mbyte sound library. The ST8000 with Sound Blaster emulation offers a joystick port, MIDI interface, and digital audio recording and playback. The ST8001 adds the Aria Synthesizer and a 512-Kbyte sound library; the ST8002 adds a 1-Mbyte sound library. Each Aria chip set contains a controller, DSP, and memory device. *Sierra Semiconductor; from $35 to $60 (10,000s).*

Reader Service No. 15

Control spectrum analyzers from afar

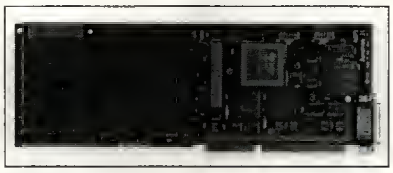
A remote personal computer user can query the MS2601 spectrum analyzer, issue commands, and receive the analyzer's responses with software via the RS-232C serial communications port. The command-driven software automatically acquires and continuously updates spectral waveforms. A reserved bottom portion of the personal computer screen accepts commands and displays the analyzer's responses. *Anritsu Wiltron.*

Reader Service No. 16

Enhanced board performs at 40 Mflops

The TMS320C30-based Banshee II System DSP add-in board for AT (ISA) bus computers works at 40 MHz to achieve 40-Mflops performance in R&D applications. Equipped with 4 Mbytes of zero-wait-state SRAM, the board uses dual-port memory for C30-to-host communications and 8-Kbyte dual-port RAMs with interrupts provided in each direction. Users with unique requirements can develop special interface boards for the Banshee II, since the host bus and the C30 memory and I/O buses are brought out to connectors. *Atlanta Signal Processors; from $3,895 (OEM board with 256-Kbyte SRAM).*

Reader Service No. 17


Banshee II DSP system from Atlanta Signal Processors

CAD software

Unprotected network EDA tools offered

A Unix version of the CapFast EDA tools permitting hierarchical schematic capture, interface, and schematic/symbol translation support Sun/Sparc, HP, and DEC workstations under an unrestricted network license. The package allows other X Window systems on the same network without restricting the number of systems accessing the CapFast software on the protected host. Users can run the same software across different X Window workstations, terminals, and PCs with X Window servicer software. *Phase Three Logic; $4,995 per host computer with $600 annual support fee (first-year support/updates free).*

Reader Service No. 18

Route Editor updated

Version 3.0 of Route Editor, the grid-free auto/interactive routing tool for high-speed, high-density designs, introduces angle-free capability and graphical min/max delay constraints routing with real-time analysis. Route Editor is part of CAD Expert within the Expert Series tool suite, which provides integrated system design methodology. This release routes orthogonally then automatically switches into free-angle mode where required, maximizing route densities and achieving completion with the fewest layers. Route Editor 3.0 can graphically visualize critical signal min/max delay constraints via a pair of graphical interactive octagons that represent the boundaries of the constraints. It is available on all company-supported Unix platforms. *Racal-Redac.*

Reader Service No. 19

Interface translates hardware into VHDL

SES/vhdl, an interface from SES/workbench, translates the hardware portion of a design into VHDL for implementation. Designers use the high-level VHDL representation as a system-level specification. The VHDL representation provides a simulation model within which other components, such as ASICs, can be evaluated during development and serves as a starting point for further refinement of the overall system design. *SES; $6,000 (on Sun workstations).*

Reader Service No. 20

View cross-section devices

IC designers and educators can program a cross-section device viewer with the enhanced L-Edit layout editor to view a simulated cutaway of the third dimension of a layout. ASIC designers can program the viewer to simulate grow/deposit, implant/diffuse, and etch steps for the layers in a mask layout. These steps simulate the corresponding steps used in the VLSI fabrication process. Boolean combinations of layers can be used, and different viewing colors and processing depths chosen. The layout editor, cross-section viewer, and optional modules run on PC, Sun/Sparc, HP-9000/400 and 700, SGI/Indigo, and Macintosh systems. *Tanner Research.*

Reader Service No. 21

Automatically synthesize networks

The Network Synthesis System automatically synthesizes application-specific computer networks, expressing network requirements as dataflow graphs with Ada or C functions as the nodes. NSS assigns programs in the application to a network of processors, provides memory and communications I/O as needed, and generates timelines and reports showing how well that application maps onto the network. As users specify constraints, NSS evaluates the success of each solution.

NSS is integrated with the Navy's Processing Graph Methodology (PGM) tools and uses a rapid prototyping technology based on modeling processors in VHDL. NSS supports concurrent hardware/software design, measuring design trade-off impacts and automatically generating an Ada compiler from VHDL machine description. *JRS Research Laboratories.*

Reader Service No. 22

PCB design software adds router

Dynamic Router, an addition to the AutoBoard System software integration to AutoCAD, moves traces, chops them into smaller pieces, reassembles them, rips them up, and then lays them down to build a design. The result is a board portraying hand-routed patterns and fewer feed-throughs. The Dynamic Router's C++ algorithm writes a general routine that affects horizontal and vertical segments. When writing a routine to resolve crossing traces, one routine covers both instances of traces. *Great SoftWestern Company; from $2,250.*

Reader Service No. 23

Design programmable logic with one product

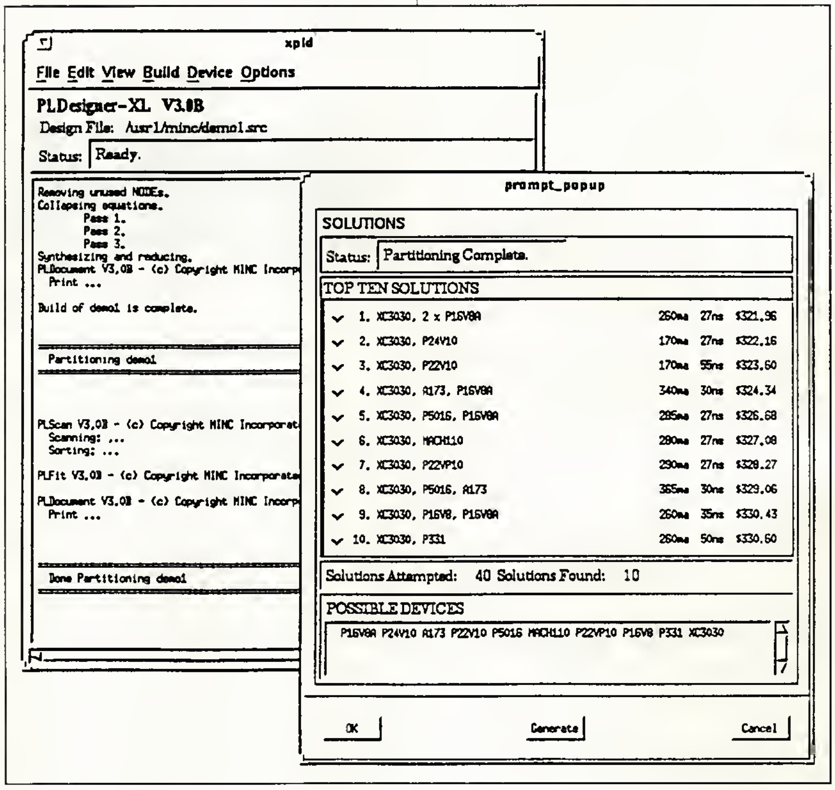
A family of synthesis tools for designing PLDs, CPLDs, and FPGAs lets users develop and implement programmable designs in a unified environment. PLDesigner-XL lets users retarget designs and try alternatives to find the best device implementation. They may use VHDL language entry or an extended version of the Minc Design Synthesis Language. A translation utility supports design imports from Abel-4 descriptions, and a schematic interface simplifies the integration of PLDesigner-XL with schematic environments.

PLDesigner-XL comes in three basic configurations: Prodigy for entry-level applications, Professional with automatic partitioning and schematic interface, and the high-end Virtuoso with VHDL input and two CPLD or FPGA device libraries. *Minc Incorporated; $2,995 (Prodigy), $4,995 (Professional), $8,995 (Virtuoso).*

Reader Service No. 24

Reduce time to emulation

A fourth-generation hardware emulation product, the RPMplus Emulation system, introduces Precision Emulation technology, which automates the mapping of nonsynchronous and asynchronous design styles. RPMplus emulates designs of up to 50,000 gates and provides X Window support, CAE design interfaces, and simulation integration. RPMplus runs on Sun, H-P/Apollo, or IBM RS/6000 workstations and supports



PLDesigner-XL V3.0B
Design File: /usr1/minc/demo1.src
Status: Ready.

```
Removing unused NODEs.
Collapsing equations.
        Pass 1.
        Pass 2.
        Pass 3.
Synthesizing and reducing.
PLDocument V3,0B - (c) Copyright MINC Incorp
  Print ...

Build of demo1 is complete.

  Partitioning demo1

PLScan V3,0B - (c) Copyright MINC Incorporat
  Scanning: ...
  Sorting: ...

PLFit V3.0B - (c) Copyright MINC Incorporate

PLDocument V3.0B - (c) Copyright MINC Incorp
  Print ...

  Done Partitioning demo1
```

SOLUTIONS
Status: Partitioning Complete.

TOP TEN SOLUTIONS

```
✓  1. XC3030, 2 x P16V8A        260ma  27ns  $321.96
✓  2. XC3030, P24V10            170ma  27ns  $322.16
✓  3. XC3030, P22V10            170ma  55ns  $323.60
✓  4. XC3030, A173, P16V8A      340ma  30ns  $324.34
✓  5. XC3030, P5016, P16V8A     285ma  27ns  $326.68
✓  6. XC3030, MACH110           280ma  27ns  $327.08
✓  7. XC3030, P22VP10           290ma  27ns  $328.27
✓  8. XC3030, P5016, A173       365ma  30ns  $329.06
✓  9. XC3030, P16V8, P16V8A     260ma  35ns  $330.43
✓ 10. XC3030, P331              260ma  50ns  $330.60
```

Solutions Attempted: 40 Solutions Found: 10

POSSIBLE DEVICES
P16V8A P24V10 A173 P22V10 P5016 MACH110 P22VP10 P16V8 P331 XC3030

MINC's PLDesigner-XL

direct data extraction for all users of Verilog and Mentor Graphics tools and simulation environments. *Quickturn Systems; $175,000.*

Reader Service No. 25

Design, test mixed-signal ICs

According to its manufacturer, an analog and mixed-signal IC, EDA design-to-test product called Dantes reduces typical development to two to three months through its tight integration with the Analog Artist Design System. The tight integration makes it possible to include test specifications early in the design cycle.

Dantes lets designers and test engineers work on a design concurrently, storing information in a format understandable and accessible to both. In addition, individual test modules can be sequenced within the automatically generated test program to allow high dropout and short duration tests to be placed first. Dantes runs under the company's Design Framework II architecture. *Cadence Design Systems; $80,000.*

Reader Service No. 26

Human interfaces

Keyboard emulates DEC VT

According to the manufacturer, the PowerStation keyboard has the exact layout of a DEC VT200/VT300 keyboard and plugs directly into the PC. Used with Zstem or KEAterm emulation software, it makes a PC, AT, or PS/2 appear and function as a VT terminal. *KEA Systems.*

Reader Service No. 27

PC support for dial-in users

The Packet/400 connectivity package for IBM AS/400 PC Support and OfficeVision now extends PC Support to Async dial-in users over X.25 packet networks. Version 2.0.0 features Gateway and Workstation inactivity timers, source-routing capabilities, inbound and like-character compression, and removable workstation software. It also includes support for communication ports 3 through 8 and 19,200 bps. The release's Workstation component emulates a Token-Ring adapter, while the Gateway component functions as a single communications controller, relaying messages among 32 simultaneous remote workstations and the AS/400. *Telepartner International.*

Reader Service No. 28

Processor increases RS/6000 performance

The RS/6000 Network Processor adapts TCP/IP networks for RISC System/6000 workstations and servers, providing high-performance connections. According to the company, using this connection lets users in client/server environments match their network bandwidth with their computer bandwidth. RS/6000 occupies a single slot and provides a full-duplex, 250-Mbps fiber optic or coaxial cable serial connection to a company networking hub. *Ultra Network Technologies; $6,500 each, volume discounts on quantity orders.*

Reader Service No. 29

Unix development tools

PowerHouse 4GL Version 7 is a client/server enhancement of the company's fourth-generation language for six commercial Unix platforms, including first-time support for SCO Unix. The recent release integrates client PCs with Unix servers.

An additional tool, PowerHouse Windows, uses an object-oriented forms repository to simultaneously support the Microsoft Windows GUI and character-based terminals with the same application. It connects clients and servers over various TCP/IP networks and serial connections. A third tool is an open, scalable architecture for transparently integrating PowerHouse tools with relational databases and networks. *Cognos Incorporated; from $1,000 to $150,000.*

Reader Service No. 30

X.400 gateway sends multimedia objects

The Mail Gateway to X.400 Version 3.0 allows users to send messages, binary files, and multimedia objects from one LAN to another across an X.400 backbone using a message encapsulation technique. It complies with the CCITT 1984 X.400 messaging standards and interoperates with DEC, Retix, and H-P products.

An added feature improves usability and shields users from X.400 complexities. This encapsulation feature makes X.400 transparent to end users; renders user names in plain English with minimal header information; and uses convenient addressing templates, aliases, and personal address books. *Microsoft Corporation; $4,995 each, $1,250 (upgrades).*

Reader Service No. 31

Macintoshes/IBMs supported

Extra! for Macintosh and 3270 Gateway Option Version 3.0 extend connectivity support to Apple Macintosh and IBM-compatible workstations. Extra! for Macintosh connects with SNA networks to communicate with the mainframe via the 3270 Gateway Option Version 3.0 on Ethernet, Token Ring, or Local Talk LANs. The emulator complies with IBM communications standards of its DOS and Windows counterparts, offering quick screen refresh, high-speed file transfers, and the

Advertisement

WANTED!
PRECIOUS METAL
ELECTRONIC SCRAP!

- Highest Returns
- Fast Turnaround

P.C. Boards • I.C.s • Connectors • Plugs

RFE INDUSTRIES
Keasbey, NJ 800-327-7938
(Over 20 Years Experience)

Macintosh look and feel.

The 3270 Gateway Option supports LAN-connected PC and PS/2-compatible workstations as well as the Macintoshes. Session-management features include NetView Alert support, event tracking, and enhanced LU-session management. *Attachmate Corporation; $425 (Extra!), $50 (Gateway Option).*

Reader Service No. 32

Link RS-422/485 with RS-232

Five user-selectable modes of operation are available with Model 285, an RS-232-to-RS-422/RS-485 interface converter called the Superverter. When configured to operate as an RS-232-to-RS-422 converter, Model 285 converts full-duplex data, TD, and RD, between the two protocols and transfers data at 64 Kbps. As an RS-232-to-RS-485 arrangement, Model 285 operates as a two-wire, half-duplex, or a 4-wire, full-duplex converter. *Telebyte Technology; $148 each (quantity discounts).*

Reader Service No. 33

Telebyte Technology's Superverter

Macintoshes run PC software

Five SoftPC emulation packages allow users to open multiple PC and Macintosh windows at the same time, cut and paste text, and transfer complete documents between applications. Entry Level SoftPC emulates basic IBM AT computers with all standard PC features. The emulator lets the Macintosh keyboard and mouse function as a 102-key AT keyboard and a Microsoft mouse, enabling users to print from PC applications directly to a Macintosh printer. Modems connected to the serial ports on a Macintosh can be used with many PC communications programs.

Universal SoftPC supports text-based applications and limited graphics programs. SoftAT with EGA graphics, a math coprocessor, and expanded memory emulation, aids graphics and intensive-calculation PC applications. Softnode interacts as a working node on an existing IBM network by providing access to the Novell Netware server. When used with Universal SoftPC or SoftAT, Softnode enables the Macintosh to share MS-DOS applications with multiple users. *Insignia Solutions.*

Reader Service No. 34

Chips

Nonvolatile memories dissipate low power

Four 3V nonvolatile memory devices extend battery life in portable equipment. They include the 1-Mbyte HN58V1001 EEPROM, 256-Kbyte HN58V257 EEPROM, 16-Mbyte HN624116L mask ROM, and 1-Mbyte HN27V101A EPROM. Sample quantities are available. *Hitachi America; from $12 to $45.*

Reader Service No. 35

Logic devices communicate with 5V systems

The Low Voltage Technology (LVT) series of logic devices run internally off a 3.3V supply and interface to both 5V and 3.3V input signals. The series consists of 8-bit octal and Widebus 16-/18-bit versions of transceivers, latches, flip-flops, and universal bus transceivers for use in high-end PC, desktop workstation, and telecom applications. The 18-bit universal bus transceiver can be configured in either transparent, latched, or registered dataflow modes. A Bus Hold feature helps designers keep the total number of component packages per board to a minimum and ensure data integrity and reliability at the input. *Texas Instruments Semiconductor Group.*

Reader Service No. 36

Nonvolatile 1-Mbit SRAM

Designated the M48Z128, the Zeropower static RAM integrates a 128K×8 CMOS SRAM, a power-fail control circuit, and a lithium battery into one 600-mil, 32-pin, plastic DIP package. The 1-Mbit device's built-in power-fail circuit constantly monitors the V_{CC} line, switching automatically to the lithium energy source whenever an out-of-tolerance condition occurs. At the same time, the chip is switched to its write-protected mode to prevent spurious write cycles. This technique allows 10-year data retention and standard SRAM access speed with equal read and write cycle times. It supports an unlimited number of write cycles. *SGS-Thomson Microelectronics; $54.60 (85 ns), $52 (120 ns) (1,000s).*

Reader Service No. 37

LCD driver extends battery life

A low-voltage CMOS IC drives LCDs in cellular telephone and remote paging systems and operates on 3V supply voltages. The e1350 typically uses 95 μA of power, enhancing battery life. Because the device can operate in direct mode as well as 4:1 multiplexing, configurations can be created using up to eight ICs in cascade to drive up to 128 characters for other types of low-power applications. *Siliconix Incorporated; $3 (OEM quantities).*

Reader Service No. 38

Reader Interest Survey

Indicate your interest in this department by circling the appropriate number on the Reader Service Card.

Low 189 Medium 190 High 191

Product Summary

Joe Hootman
University of North Dakota

Manufacturer	Model	Comments	R.S.#
Boards			
Micro Energy	SPP040-2FNN DC/DC converter	Dual-output DC/DC module converts 36- to 60-VDC input to 5- and 12-VDC output using 40/65W at peak performance. The 4×1.5×0.562-inch converter in thermal metal-clad board and surface mount technology offers 1,000,000-MTBF reliability and includes self-contained heat sinking, remote on/off, over-voltage and over-current protection.	80
New Media	Publish Cards flash IC cards	Credit card-size line fits in subnotebook and palmtop PCs equipped with a PCM-CIA interface slot. Capable of storing 256 Kbytes to 100 Mbytes of data, the line is suited to use in rugged environments. $10 per megabyte.	81
Philips Semiconductors	TDA5145 motor controller	Integrated driver IC for three-phase, brushless DC motors provides 1.8A direct drive to the motor windings without requiring a Hall-effect sensor motor. Suited for hard-disk and tape drives, the controller features bidirectional rotation and braking facilities. $2 in volume quantities, depending on importing country.	82
Software			
Legacy Systems	Convert2 unit converter	Program on 3.5 or 5.25 disk converts scientific/engineering units for users with PCs, Windows 3.0, and a hard disk with 400 Kbytes of free space. Convert2 contains an editor activated by touching a button on the action bar, a calculator, and a help feature. $19.95 plus $3 shipping/handling in US, $5 outside US.	83
Ultimedia	MMPM/2 manager	IBM Multimedia Presentation Manager/2 extends the Operating System/2 2.0 to add audio capabilities to 32-bit PCs. MMPM/2 supports digital audio file playback and audio signal recording into digital format using the IBM M-Audio Capture and Playback Adapter/A. An optional tool kit (MMPMTK/2) contains C language bindings, sample programs and documentation, and assists in the use of MMPM/2. $125; $199 for tool kit on CD-ROM.	84
Miscellaneous			
Bit 3 Computer	Model 350-010 switch	Shared-memory switch for the RISC System/6000, when used with ParaSoft's Express software, functions as a stand-alone communication hub to accelerate parallel processing applications with a cluster of four RS/6000s. Model 350-010 consists of a VME cabinet with 8 Mbytes of VMEbus memory, an arbiter card, AIX device driver, and AIX diagnostic software. Up to four of the company's VMEbus Adaptor cards can be installed, which interconnect to the RS/6000 systems. $4,995; $2,850 (Adaptor).	85

FOR DISPLAY ADVERTISING INFORMATION, CONTACT:

Southern California and Mountain States: Richard C. Faust, Douglas C. Faust, 24050 Madison Street, Suite 101, Torrance, California 90505; Tel: (310) 373-9604; Fax: (310) 373-8760.

East Coast: Gail A. Frank, Nancy Inserra, 82 Bethany Road, Suite 4, Hazlet, New Jersey 07730; Tel: (908) 264-1100; Fax: (908) 264-4340.

New England: Paul Gillespie, PO Box 6444, Holliston, Massachusetts 01746; Tel: (508) 429-8907; Fax (508) 42

9-3285.

Advertising Manager: Heidi Rex, 10662 Los Vaqueros Circle, Los Alamitos, California 90720-1264; Tel: (714) 821-8380; Fax: (714) 821-4010.

For production information, conference, and classified advertising, contact Marian Tibayan.

IEEE MICRO, 10662 Los Vaqueros Circle, PO Box 3014, Los Alamitos, California 90720-1264; phone (714) 821-8380; fax (714) 821-4010.

	RS #	Page #
Anritsu Wiltron	16	75
Atlanta Signal Processors	17	75
Attachmate Corp.	32	78
Bit 3 Computer	85	79
Cadence Design Systems	26	77
Cognos Inc.	30	77
Great SoftWestern Co.	23	76
Hitachi America	35	78
Insignia Solutions	34	78
JRS Research Laboratories	22	76
KEA Systems	27	77
Legacy Systems	83	79
Markenrich Corp.	13	74
Micro Energy	80	79
Microsoft Corp.	31	77
Minc Inc.	24	76
New Media	81	79
Phase Three Logic	18	75
Philips Semiconductors	82	79
Quickturn Systems	25	77
Racal-Redac	19	75
SES	20	75
SGS-Thomson Microelectronics	37	78
Sierra Semiconductor	15	75
Siliconix Inc.	38	78
Spectrum Signal Processing	12, 14	74, 75
Tanner Research	21	76
Telebyte Technology	33	78
Telepartner International	28	77
Texas Instruments	11, 36	74, 78
Ultimedia	84	79
Ultra Network Technologies	29	77

IEEE COMPUTER SOCIETY®

A member society of the Institute of Electrical and Electronics Engineers, Inc.

THE FOLLOWING INFORMATION IS AVAILABLE:

Contact the Publications Office;
to facilitate handling, please request by number.

- Membership application, student #203, others #202
- Periodicals subscription form for individuals #200
- Periodicals subscription form for organizations #199
- Publications catalog #201
- Compmail electronic mail brochure #194
- Technical committee list/application #197
- Chapters lists, start-up procedures #193
- Student scholarship information #192
- Volunteer leaders/staff directory #196
- IEEE senior member grade application #204
  (requires ten years practice and significant performance in five of those ten)

To check membership status or report a change of address, call the IEEE toll-free number, 1-800-678-4333. Direct all other Computer Society-related questions to the Publications Office.

PURPOSE

The IEEE Computer Society advances the theory and practice of computer science and engineering, promotes the exchange of technical information among 100,000 members worldwide, and provides a wide range of services to members and nonmembers.

MEMBERSHIP

Members receive the acclaimed monthly magazine *Computer*, discounts, and opportunities to serve (all activities are led by volunteer members). Membership is open to all IEEE members, affiliate society members, and others interested in the computer field.

PUBLICATIONS AND ACTIVITIES

Computer. An authoritative, easy-to-read magazine containing tutorial and in-depth articles on topics across the computer field, plus news, conferences, calendar, interviews, and product reviews.

Periodicals. The society publishes seven magazines and five research transactions. Refer to membership application or request information as noted at left.

Conference Proceedings, Tutorial Texts, Standards Documents. The Computer Society Press publishes more than 100 titles every year.

Standards Working Groups. More than 100 of these groups produce IEEE standards used throughout the industrial world.

Technical Committees. More than 30 TCs publish newsletters, provide interaction with peers in specialty areas, and directly influence standards, conferences, and education.

Conferences/Education. The society holds about 100 conferences each year and sponsors many educational activities, including computing science accreditation.

Chapters. Regular and student chapters worldwide provide the opportunity to interact with colleagues, hear technical experts, and serve the local professional community.

OMBUDSMAN

Members experiencing problems — magazine delivery, membership status, or unresolved complaints — may write to the ombudsman at the Publications Office.

IEEE MICRO

Editorial Calendar

FEBRUARY 1993

Automotive/traffic microelectronics
- Worldwide developments in microelectronics for traffic and driving assistance
- Improving traffic safety with electronics
- Latest developments from Japan, the European Prometheus, and US IVHS programs

Ad closing date: January 2

AUGUST 1993

Multitheme issue
- Multiprocessing
- Optical and biological computing
- Microcomputing to aid the handicapped
- Systems design
- DSP-based tools

Ad closing date: July 1

APRIL 1993

Advanced packing and interconnect technology
- Critical packaging trends and issues
- Substrate and package technologies—for example, flexiblc, glass, or diamond substrates; few-chip or 3D packaging
- Attachment, bonding, and connection technologies, including fine-pitch surface mount, laser applications, known-good die, and interconnection trade-off analysis

Ad closing date: March 1

OCTOBER 1993

Far East issue
- Microcomputer and RAM technology
- High-performance parallel computer research
- Current TRON Project offerings, Japan's standard computer system
- Singapore technology update

Ad closing date: September 1

JUNE 1993

Hot Chips IV
- This extremely popular issue presents the latest developments in microprocessor and chip technology used to construct high-performance workstations and systems as presented at the annual IEEE Computer Society TCMM-sponsored symposium

Ad closing date: May 1

DECEMBER 1993

Special standards issue
- Buses and interconnections
- Processor architectures
- Current activities: IEEE and other standards bodies
- How standards can help designers

Ad closing date: November 1

IEEE Micro helps designers and users of microprocessor and microcomputer systems explore the latest technologies to achieve business and research objectives. Feature articles in *IEEE Micro* reflect original works relating to the design, performance, or application of microprocessors and microcomputers. All manuscripts are subject to a peer-review process consistent with professional-level technical publications. *IEEE Micro* is a bimonthly publication of the IEEE Computer Society.

Advertising information: Contact Marian Tibayan, Advertising Department, IEEE Computer Society, PO Box 3014, Los Alamitos, CA 90720-1264; (714) 821-8380; fax (714) 821-4010.

Articles may change. Please contact editor to confirm.